HEWLETT® PACKARD

HP PCL/PJL Reference

PCL 5 Printer Language Technical Reference Manual

Part I

Click here to access Part II on hp.com.

PCL 5 Printer LanguageTechnical Reference Manual

HP Part No. 5961-0509
Printed in USA
First Edition - October 1992

Notice

Printing History

This manual was created using *HP Tag/Vectra* software on an HP Vectra Personal Computer. The body text is printed in *Helvetica* fonts. The camera-ready copy was printed on an HP LaserJet IIISi printer with Resolution Enhancement technology (REt) and was then reproduced using standard offset printing.

First Edition – October 1992

NOTICE

This document is the current edition of the technical reference manual for PCL 5 and earlier printers. It replaces the September 1990 edition of the *HP PCL 5 Printer Language Technical Reference Manual* (p/n 33459-90903). If you have ordered another PCL Technical Reference document, this manual and the *PCL 5 Comparison Guide* are the updated replacement documents.

Trademark Credits

Intellifont and *Garth Graphic* are U.S. registered trademarks of Agfa Division, Miles Inc. *CG Triumvirate* and *Shannon* are trademarks of Agfa Division, Miles Inc. *CG Bodoni*, *CG Century Schoolbook*, *CG Goudy Old Style*, *CG Melliza*, *Microstyle*, *CG Omega*, *CG Palacio*, *CG Times* and *CG Trump Mediaeval* are products of Agfa Division, Miles Inc. *CG Times*, a product of Agfa Division, Miles Inc., is based on *Times New Roman*, a U.S. registered trademark of Monotype Corporation plc. *PCL*, and *Vectra* are U.S. registered trademarks of Hewlett-Packard Company. *Resolution Enhancement* is a trademark of Hewlett-Packard Company. *IBM* is a registered trademark of International Business Machines Corporation. *Microsoft*, *Windows*, and *MS-DOS* are U.S. registered trademarks of Microsoft Corporation. *TrueType* and *Macintosh* are registered trademarks of Apple Computer, Inc. *PostScript* is a registered trademark of Adobe Systems, Inc. *Centronics* is a U.S. registered trademark of Centronics Corporation. *ITC Avant Garde Gothic*, *ITC Benguiat*, *ITC Bookman*, *ITC Cheltenham*, *ITC Galliard*, *ITC Korinna*, *ITC Lubalin Graph*, *ITC Souvenir*, *ITC Zapf Chancery* and *ITC Zapf Dingbats* are U.S. registered trademarks of International Typeface Corporation. *ITC Tiffany* is a trademark of International Typeface Corporation. *Futura* is a U.S. registered trademark of Fundicion Tipografica Neufville, S.A. *Serifa* is a trademark of Fundicion Tipografica Neufville, S.A. *Letraset* is a registered trademark of Esselte Pendaflex Corporation. Revue and *University Roman* are trademarks of Esselte Pendaflex Corporation. *Helvetica* and *Times Roman* are trademarks of Linotype AG and its subsidiaries. *Univers* is a U.S. registered trademark of Linotype AG and its subsidiaries. *Antique Olive* is a trademark of Monsieur Marcel OLIVE. *Arial* and *Gill Sans* are registered trademarks of The Monotype Corporation plc.

Inside This Manual

What You Can Learn From This Manual

Hewlett-Packard has developed a standard set of printer features for use in all HP printers. Printer features are accessed through the corresponding commands of Hewlett-Packard's PCL language. This manual describes the PCL 5 printer language. This includes descriptions of the commands available for Hewlett-Packard PCL 5 LaserJet printers and the basic requirements of PCL language programming. With the release of new LaserJet family printers there are new features added which supplement the existing PCL base set. Features of future printer releases are not covered in this document. The new features are described in the latest version of the *PCL 5 Comparison Guide*. Programmers should familiarize themselves with the information provided in the *PCL 5 Comparison Guide* in addition to the information in this document.

Experienced Users

This manual was written for people with some programming experience. Many of the concept discussions assume some programming knowledge.

When writing a PCL language program, you should know the PCL language concepts and commands presented in this manual, and should be aware of the differences in implementation of the PCL 5 printer language for the various HP LaserJet printers, as described in the *PCL 5 Comparison Guide*.

Non-technical Users

Many software applications (word processing software, spreadsheets, etc.) allow you to embed printer commands as **escape sequences** in the body of your documents. This manual presents the full syntax and explanation of all the commands supported by PCL 5 LaserJet printers. These commands enable you to take advantage of the LaserJet printer's advanced feature set.

Note Since actual implementation of printer commands within software applications varies from package to package, specific examples are not given. For examples of printer command usage with many popular software packages, refer to HP's *Software Application Notes*, provided with the printer. The most current versions of software application notes can be obtained through the HP Forum on CompuServe, by fax using the HP FIRST fax service, or through HP's literature distribution. Refer to Appendix A for more information.

Chapter Summaries

A brief description of each chapter is provided below.

Chapter 1 - Introduction to HP PCL

This chapter gives a brief history of the development of the PCL language, describes the PCL language levels (architecture), and describes the PCL command structure (control codes and escape sequences).

Chapter 2 - The Page

This chapter introduces the idea of the logical page and identifies the area in which printing can occur. It also describes the PCL coordinate system and the HP-GL/2 picture frame.

Chapter 3 - The Print Environment

This chapter introduces the printer's feature settings, collectively, as the print environment. It includes descriptions of the factory default environment, user default environment, and the modified print environment. The effect of printer reset functions is also described.

Chapter 4 - PCL Job Control Commands

This chapter describes the commands which provide job control. Job control commands are usually grouped together and sent at the beginning of a job. Job control includes restoration of the User Default Environment, selection of the number of copies of each page to be printed, duplex print commands, and unit of measure specification.

Chapter 5 - Page Control Commands

This chapter describes the commands providing page format control. Page format control allows you to select the page source, size, orientation, margins, and text spacing.

Chapter 6 - Cursor Positioning

This chapter describes how to position the cursor within the logical page.

Chapter 7 - Fonts

This chapter describes basic font information including font characteristics.

Chapter 8 - PCL Font Selection

This chapter describes how to select a font for printing using the font characteristics commands. The underline feature is described at the end of the chapter.

Chapter 9 - Font Management

This chapter describes font management which provides mechanisms for downloading and manipulating soft fonts.

Chapter 10 - User-Defined Symbol Sets

This chapter describes the capability of some PCL 5 printers to enable users to define their own symbol sets for special needs.

Chapter 11 - Soft Font Creation

This chapter describes how to organize font/character data for downloading to the printer.

Chapter 12 - Macros

This chapter describes macro commands which store a block of PCL commands and data which can be used repeatedly without redefining the block. The macro function reduces the number of commands that must be sent to the printer.

Chapter 13 - The PCL Print Model

This chapter describes the PCL print model which allows for special effects when printing.

Chapter 14 - PCL Rectangular Area Fill Graphics

This chapter describes how to define and fill a rectangular area with one of the predefined PCL patterns, or with a user-defined pattern.

Chapter 15 - Raster Graphics

This chapter describes how to download raster graphics to the printer, and includes various techniques for reducing the amount of data needed to define the raster image.

Chapter 16 - Status Readback

This chapter describes the PCL status readback features. Status readback enables you to obtain PCL status information from the printer, such as: available printer (user) memory, a list of fonts and symbol sets, and the ID numbers of macros and user-defined patterns.

Chapter 17 - An Introduction to HP-GL/2 Vector Graphics

This chapter introduces basic information for HP-GL/2. It lists the vector graphics commands, and describes the HP-GL/2 command syntax. An overview of several important topics is also provided, such as the PCL Picture Frame concept, scaling, pen status and location, and absolute vs. relative pen movement.

Chapter 18 - The Picture Frame

This chapter describes how to set up an area on the page for printing vector graphics (the PCL Picture Frame). It discuses the commands necessary to define and position the picture frame, along with the commands used to enter and exit HP-GL/2 mode.

Chapter 19 - The Configuration and Status Group

This chapter describes the commands used to set default conditions and values for programmable HP-GL/2 features. It also explains the commands used for scaling, establishing a soft-clip window, and rotating the HP-GL/2 coordinate system.

Chapter 20 - The Vector Group

This chapter provides information about pen movement and drawing lines, arcs, and circles. It also covers a way to encode coordinates for increased print speed.

Chapter 21 - The Polygon Group

This chapter explains the polygon mode and how it is used to draw polygons, subpolygons and circles. The commands for drawing and filling wedges and rectangles are also described in this chapter.

Chapter 22 - The Line and Fill Attributes Group

This chapter describes the commands used to vary the line types and fill patterns used to create HP-GL/2 graphics.

Chapter 23 - The Character Group

This chapter contains information about the commands used to print text (labels) in HP-GL/2 mode. This allows you to print HP-GL/2 labels in almost any size, slant and direction using proportional or fixed-spaced scalable fonts.

Chapter 24 - Programming Hints

This chapter provides programming information for use during the development of PCL software.

Related Documentation

The following related manuals provide further information about HP LaserJet printers, including their features and functions.

PCL 5 Comparison Guide

This document contains supplemental information for programming PCL 5 LaserJet printers. It identifies how different HP PCL 5 LaserJet printers implement the commands described in the *HP PCL 5 Printer Language Technical Reference Manual*. It provides printer-specific information on feature sets, paper handling, fonts, and the printer's control panel.

Intellifont Scalable Typeface Format

This document provides information for designing scalable fonts using Agfa's Font Access Interchange Standard (FAIS). This document can be obtained from Agfa Division, Miles Inc. by writing to the address below or by phone.

Agfa Division, Miles Inc.
Typographic Systems
OEM Technical Support
90 Industrial Way
Wilmington, MA 01887
(508) 658-5600

TrueType Font Files

This document, which provides information for designing scalable fonts using Microsoft Corporation's TrueType font scaling technology has been made available in downloadable form on both CompuServe and Internet. Contact Microsoft Corporation for details.

Contents

Introduction to HP PCL

The Page

The Print Environment

PCL Job Control Commands

Page Control Commands

Cursor Positioning

Fonts

PCL Font Selection

Font Management

User-Defined Symbol Sets

Soft Font Creation

Macros

The PCL Print Model

PCL Rectangular Area Fill Graphics

Raster Graphics

Status Readback

An Introduction to HP-GL/2 Vector Graphics

The Picture Frame

The Configuration and Status Group

The Vector Group

The Polygon Group

The Line and Fill Attributes Group

The Character Group

Programming Hints

Customer Support

Index

1 Introduction to HP PCL

PCL PRINTER LANGUAGE HISTORY

Hewlett-Packard created the PCL printer language (simply referred to as “PCL” elsewhere in this manual) to provide an economical and efficient way for application programs to control a range of printer features across a number of printing devices. HP has evolved both the definition and implementations of PCL to provide the optimal price and performance balance. PCL 5 represents a new breakthrough in price/performance leadership. Its features were selected in direct response to customer requests. HP will continue to lead enhancements to the PCL printer language to deliver powerful technology advances.

PCL commands are compact escape sequence codes that are embedded in the print job data stream. This approach minimizes both data transmission and command decoding overhead. HP PCL formatters and fonts are designed to quickly translate application output into high-quality, device-specific, raster print images.

PCL printer language commonality from HP printer to HP printer helps to minimize printer support problems and protect HP printer customer investment in applications and printer driver software.

PCL Printer Language Architecture

PCL printer language structure has been useful to guide language functionality growth and command syntax definition. The PCL printer language has evolved through five major levels of functionality driven by the combination of printer technology developments, changing user needs, and application software improvements. The five phases of the PCL printer language evolution are:

PCL 1	*Print and Space* functionality is the base set of functions provided for simple, convenient, single-user workstation output.
PCL 2	*EDP (Electronic Data Processing) /Transaction* functionality is a superset of PCL 1. Functions were added for general purpose, multi-user system printing.
PCL 3	*Office Word Processing* functionality is a superset of PCL 2. Functions were added for high-quality, office document production.
PCL 4	*Page Formatting* functionality is a superset of PCL 3. Functions were added for new page printing capabilities.
PCL 5	*Office Publishing* functionality is a superset of PCL 4. New publishing capabilities include font scaling and HP-GL/2 graphics.

The PCL printer language model succeeds because the following points are observed:

- All HP LaserJet printers implement PCL printer language features consistently.
- HP printers implement the above language feature groups in very cost-effective formatters.
- HP printers have the ability to ignore most unsupported commands.

What are Printer Commands?

PCL **printer commands** provide access to printer features. There are four general types of HP printer language commands:

- control codes
- PCL commands
- HP-GL/2 commands
- PJL commands

Control Codes

A control code is a character that initiates a printer function, for example Carriage Return (CR), Line Feed (LF), Form Feed (FF), etc.

PCL Commands

PCL commands provide access to the printer's PCL control structure. The PCL structure controls all of the printer's features except those used for vector graphics, which are controlled by the HP-GL/2 commands.

PCL printer commands consist of two or more characters. The first character is always the ASCII escape character, identified by the ^{E}C symbol. ^{E}C is a special control code which identifies the subsequent string of characters as a printer command. As the printer monitors incoming data from a computer, it "looks" for this character. When this character appears, the printer reads it and its associated characters as a command to perform and not as data to print.

Note PCL printer commands (other than single-character control codes) are also referred to as **escape sequences**. The terms *printer command* and *escape sequence* are used interchangeably throughout this manual.

Once a PCL command sets a parameter, that parameter remains set until that PCL command is repeated with a new value, or the printer is reset to its user default environment. For example, if you send the printer a command to set line spacing to 3 lines/inch, each page prints 3 lines/inch until the printer receives a different Line Spacing command, or the printer is reset.

HP-GL/2 Commands

HP-GL/2, vector graphic commands are two letter mnemonic codes designed to remind you of the function name (such as **IN** for Initialize). Following the two letter mnemonic may be one or more parameters, which identify details of how to process the command. For additional information on HP-GL/2 commands, refer to Chapters 17 through 23.

PJL Commands

PJL (Printer Job Language) commands provide a different type of printer control. Unlike PCL and HP-GL/2, which control the placement of dots on the printed page, PJL supplies job-level control. One of the main features PJL offers is the ability to switch printer languages (personalities) between jobs. For example, applications supporting PJL can print one job using PCL, and then print the next job using PostScript or another printer language, without any operator intervention.

PJL also provides two-way communications with the printer. For example, PJL can request information from the printer such as printer model, configuration, printer status, and job status. PJL also can be used to change the printer's control panel settings and modify the message displayed on the control panel.

The PJL language is designed to be used by application developers and technical support personnel only.

Syntax of Escape Sequences

There are two forms of PCL escape sequences: two-character escape sequences and parameterized escape sequences.

Two-Character Escape Sequences

Two-character escape sequences have the following form:

$^{E}{}_{C}$ *X*

where "*X*" is a character that defines the operation to be performed. "*X*" may be any character from the ASCII table within the range 48-126 decimal ("0" through "~" - see Appendix A). For a list of the two-character escape sequences supported by the printer, refer to the "PCL Feature Support Matrix" in Chapter 1 of the ***PCL 5 Comparison Guide***.

Following are examples of two-character escape sequences:

$^{E}{}_{C}$ **E**	a two-character escape sequence used for resetting the printer.
$^{E}{}_{C}$ **9**	a two-character escape sequence used for resetting the left and right margins to the printer's default settings.

Parameterized Escape Sequences

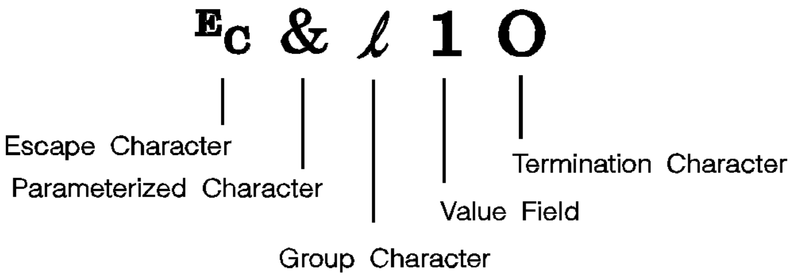
Parameterized escape sequences have the following form:

^{E}C *X y z1 # z2 # z3 ... # Zn[data]*

where *y, #, zi* (*z1, z2, z3...*) and *[data]* may be optional, depending on the command.

X	**Parameterized Character** - A character from the ASCII table within the range 33-47 decimal ("**!**" through "**/**") indicating that the escape sequence is parameterized.
y	**Group Character** - A character from the ASCII table within the range 96-126 decimal (" **'** " through " **~** ") that specifies the group type of control being performed.
#	**Value Field** - A group of characters specifying a numeric value. The numeric value is represented as an ASCII string of characters within the range 48-57 decimal ("0" through "9") that may be preceded by a "+" or "—" sign and may contain a fractional portion indicated by the digits after a decimal point (" **.** "). Numeric value fields are within the range -32767 to 65535. If an escape sequence requires a value field and a value is not specified, a value of zero is assumed.
zi	**Parameter Character** - Any character from the ASCII table within the range 96-126 decimal (" **'** " through " **~** "). This character specifies the parameter to which the previous value field applies. This character is used when combining escape sequences.
Zn	**Termination Character** - Any character from the ASCII table within the range 64-94 decimal (" **@** " through " **^** "). This character specifies the parameter to which the previous value field applies. This character terminates the escape sequence.
[data]	**Binary Data** is eight-bit data (for example, graphics data, downloaded fonts, etc.). The number of bytes of binary data is specified by the value field of the escape sequence. Binary data immediately follows the terminating character of the escape sequence.

The following is an example of an escape sequence with a termination character and no parameter character. This escape sequence performs a single function.

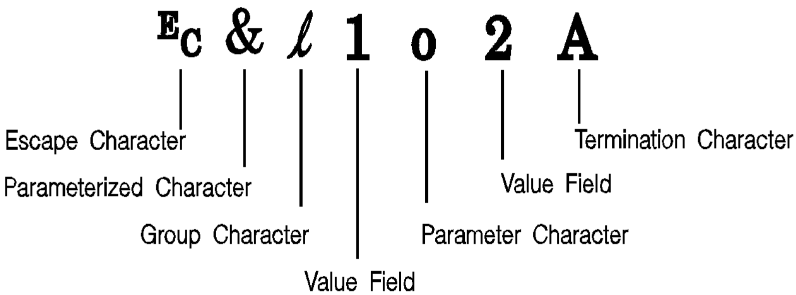


Notes

Some escape sequences shown in this manual contain spaces between characters for clarity. Do not include these spaces when using escape sequences.

Also, in the escape sequence a script "ℓ" is used to indicate a lower case "l" for clarity.

The following is an example of an escape sequence with a parameter character and a termination character. This escape sequence performs two functions. It is the combination of two commands (Ec&ℓ1O and Ec&ℓ2A):



Notice that the "Ec"and the "&ℓ"are dropped from the second printer command when the two commands are combined. Also, the upper-case "**O**" that terminated the first command becomes a lower-case "o" parameter character when these commands are combined.

Use these three rules to combine and shorten printer commands:

1. The first two characters after "^{E}C" (the parameterized and group character) must be the same in all of the commands to be combined. In the example above, these are "&" and "ℓ".

2. All alphabetic characters within the combined printer command are lower-case, except the final letter which is always upper-case. In the combined example above, "**O**" becomes "**o**". The final character in the printer command must always be upper-case to let the printer know that the command is complete.

3. The printer commands are performed in the order that they are combined (from left to right). Be sure to combine commands in the order that they are to be performed.

2 The Page

Introduction

This chapter describes the PCL coordinate system. It defines the logical page and the printable area; it introduces the HP-GL/2 (vector graphics) picture frame, and identifies the boundaries of each.

Logical Page

The PCL **logical page** (also referred to as the PCL addressable area) defines the area in which the PCL cursor can be positioned. Although the printer does not actually have a cursor (like the blinking underline character used on most computer terminals), the cursor position refers to the Currently Active Position of the cursor (also referred to as the **CAP**). The location of the "cursor" is the position on the logical page where the next character will be positioned. The cursor can be moved to different points on the logical page using the cursor positioning commands (see Chapter 6, *Cursor Positioning*). The PCL cursor cannot be moved outside of the logical page bounds.

The size of the logical page for the media (paper, transparencies, labels, etc.) is defined in Table 2-1 and Table 2-2.

Printed Dots

The high quality output achieved by HP LaserJet printers is due in part to the ability to lay down a fine grid of “dots” on the page. The density of this grid is referred to as the printer’s **resolution.** From the first HP LaserJet (the “LaserJet Classic”) until recently, all HP LaserJet family printers printed at a resolution of 300 dots-per-inch. In a one inch square, the printer could print a dot anywhere in a grid of up to 300 dots horizontally by 300 dots vertically, for a total of 90,000 possible dot locations per square inch (300 × 300 = 90,000).

The LaserJet 4 printer is capable of printing at either 300 or 600 dpi resolution. At 600 dots-per-inch, it becomes possible to print up to 360,000 dots per square inch (600 x 600 = 360,000). Print resolution of LaserJet 4 is selectable and can be specified either from the printer’s control panel or programmed through PJL commands.

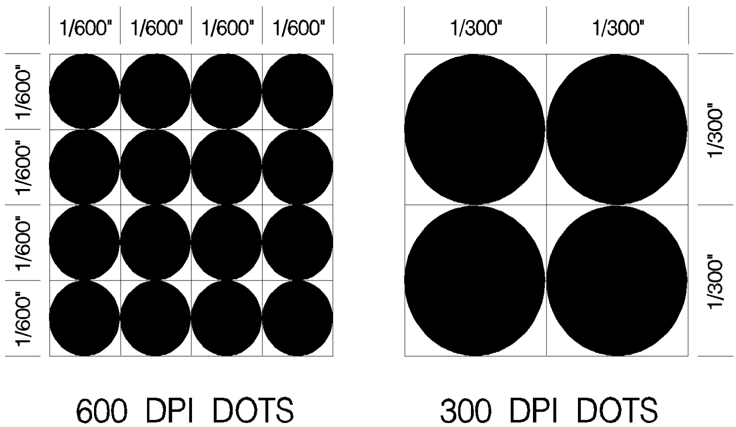


Figure 2-1 300 vs 600 DPI Dot Sizes

Note Refer to Appendix E of the *PCL 5 Comparison Guide* or the printer *User’s Manual* to determine the default print resolution for a specific HP LaserJet printer.

The printer’s physical dot size has no direct bearing on the size of “PCL Units” used in cursor movements. PCL Units were previously referred to as “PCL dots”, but **should not be confused with the printer’s physically printed dots.** The size of PCL Units can also be specified (see the *Unit of Measure Command* in Chapter 4 for more information).

PCL Coordinate System

The PCL coordinate system is defined as shown in Figure 2-2.

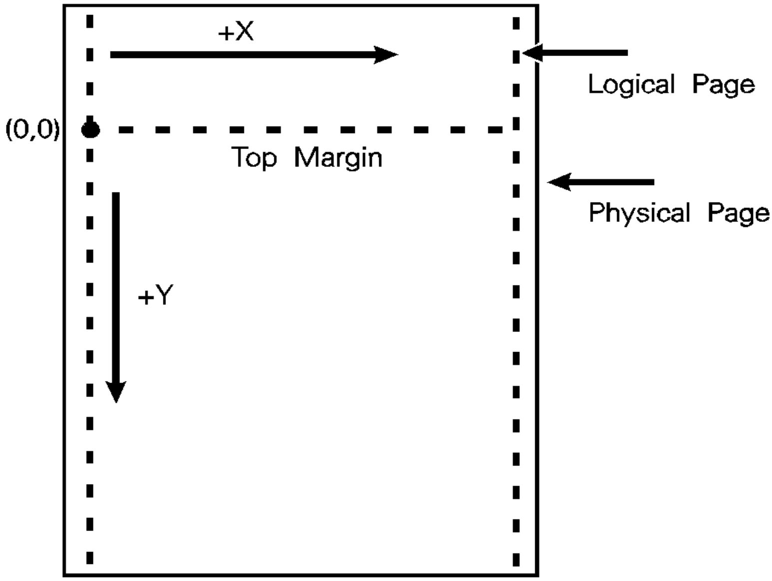


Figure 2-2 X,Y PCL Coordinates

The point (0,0) is at the intersection of the left edge of the logical page and the current top margin position.

Note Since the point (0,0) is always at the intersection of the left edge of the **logical page** and the **current top margin** position, it moves if the top margin is changed, and rotates around the page if the orientation is changed.

Units of the PCL Coordinate System

The units of the X-axis of the PCL coordinate system may be **PCL Units, decipoints**, or **columns**. The units of the Y-axis may be PCL Units, decipoints, or **rows**.

PCL Units

These are user-definable units of measure which are used in PCL commands affecting various PCL cursor moves. The number of units-per-inch used in PCL cursor moves is determined by the current setting of the **Unit of Measure** command (see “Unit of Measure Command” in Chapter 4).

Note PCL Units were formerly referred to as “PCL Dots”. They were renamed “PCL Units” to prevent confusion with the printer’s physically printed “dots”, which are determined by the printer’s resolution.

Decipoints

In PCL terminology, a decipoint is 1/720 inch or one-tenth of a PCL point (a PCL point is 1/72 inch as opposed to a typographic point which is 1/72 inch).

Columns & Rows

The width of a column is defined by the current **horizontal motion index (HMI)**. The distance between rows is defined by the current **vertical motion index (VMI)**, or **lines-per-inch (lpi)**. HMI, VMI and lpi are described in Chapter 5, *Page Control Commands*.

Printer Internal Units

Internally, the printer uses a different unit of measure. It maps PCL Units, decipoints, and columns and rows to this unit of measure. This internal unit is 1/7200 inch. All positioning is kept in internal units and rounded to physical dot positions when data is printed.

HP-GL/2 Picture Frame

In addition to text and raster graphics, **HP-GL/2 vector graphics** can be placed on the PCL logical page. HP-GL/2 vector graphics are incorporated using the concept of the HP-GL/2 picture frame (see Figure 2-3). Within this picture frame, HP-GL/2 uses its own coordinate system and units of measure. The HP-GL/2 coordinate system and units are described in detail in Chapter 17, *An Introduction to HP-GL/2 Vector Graphics*, and Chapter 18, *The Picture Frame*.

Note

If no HP-GL/2 picture frame size is specified (using the commands described in Chapter 18), then the default HP-GL/2 picture frame is used. The default HP-GL/2 picture frame is the current top and bottom margins and the left and right edges of the logical page. The HP-GL/2 picture frame rotates with the PCL page orientation, but is not affected by the PCL print direction.

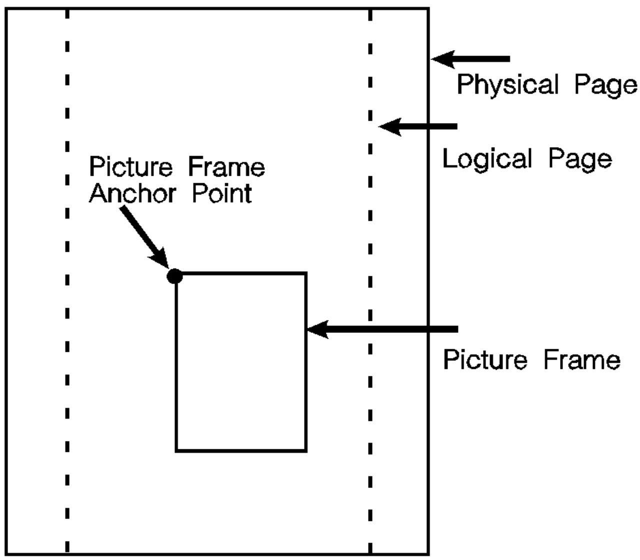


Figure 2-3 PCL Logical Page with HP-GL/2 Picture Frame

Printable Area

The **printable area** is the area of the physical page in which the printer is able to place a dot. The **physical page** refers to the size of the media (letter, legal, etc.) installed in the printer.

The relationship between physical page, logical page, default picture frame, and printable area is defined in Table 2-1 and Table 2-2.

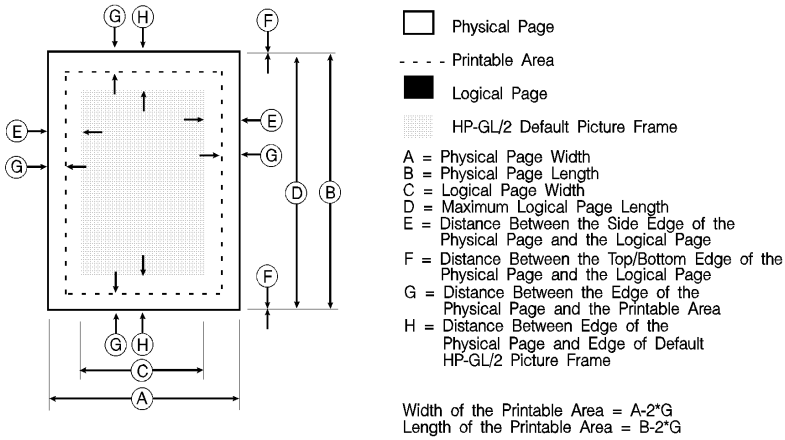


Table 2-1 Portrait Logical Page & Printable Area Boundaries

	DIMENSIONS (at 300 DPI - double for 600 DPI)							
PAPER SIZE	A	B	C	D	E	F	G	H
LETTER	2550	3300	2400	3300	75	0	50	150
Legal List 1	2550	4200	2400	4200	75	0	50	150
LEDGER	3300	5100	3150	5100	75	0	50	150
EXECUTIVE	2175	3150	2025	3150	75	0	50	150
A4	2480	3507	2338	3507	71	0	50	150
A3	3507	4960	3365	4960	71	0	50	150
COM-10	1237	2850	1087	2850	75	0	50	150
MONARCH	1162	2250	1012	2250	75	0	50	150
C5	1913	2704	1771	2704	71	0	50	150
B5	2078	2952	1936	2952	71	0	50	150
DL	1299	2598	1157	2598	71	0	50	150

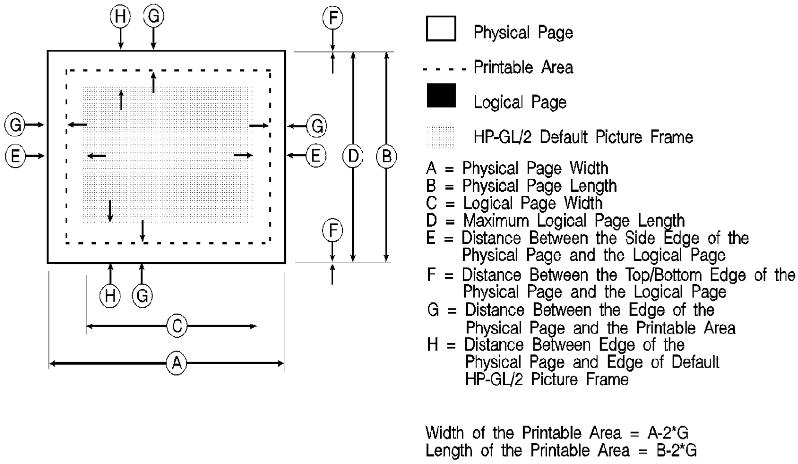


Table 2-2 Landscape Logical Page & Printable Area Boundaries

	DIMENSIONS (at 300 DPI - double for 600 DPI)							
PAPER SIZE	A	B	C	D	E	F	G	H
LETTER	3300	2550	3180	2550	60	0	50	150
Legal List 1	4200	2550	4080	2550	60	0	50	150
LEDGER	5100	3300	4980	3300	60	0	50	150
EXECUTIVE	3150	2175	3030	2175	60	0	50	150
A4	3507	2480	3389	2480	59	0	50	150
A3	4960	3507	4842	3507	59	0	50	150
COM-10	2850	1237	2730	1237	60	0	50	150
MONARCH	2250	1162	2130	1162	60	0	50	150
C5	2704	1913	2586	1913	59	0	50	150
B5	2952	2078	2834	2078	59	0	50	150
DL	2598	1299	2480	1299	59	0	50	150

The HP LaserJet printers perform pixel-level clipping. When printing characters or graphics, if any portion of the character cell or graphic is outside the printable area, only that portion outside the printable area is clipped (see Figure 2-4).

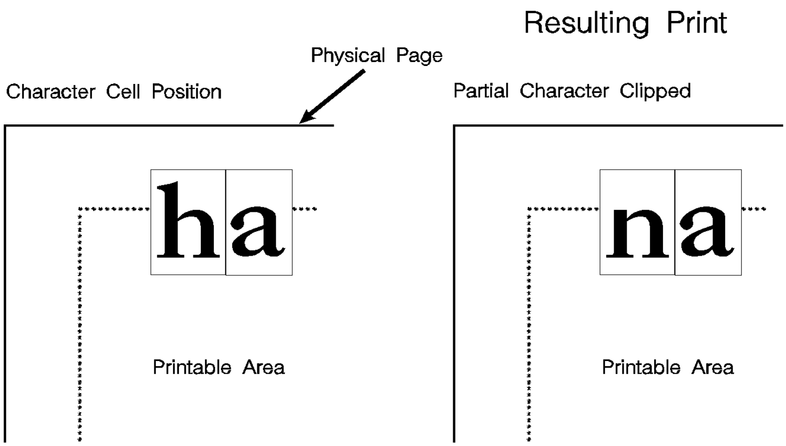


Figure 2-4 Printable Area Character Cell Positioning

Notes

Pixel level clipping can also occur at the logical page for PCL when the page is positioned using the Left Registration command.

Pixel level clipping also occurs at the picture frame for HP-GL/2.

Characters are clipped if they fall across a margin (left, right, top, and bottom). Refer to "Text Area" in Chapter 5 for additional information.

3 The Print Environment

Introduction

The group of all of the printer's current feature settings, collectively, is referred to as the print environment. The printer maintains four print environments: the **Factory Default Environment**, the **User Default Environment**, the **Modified Print Environment** and the **Overlay Environment**. This chapter describes the Factory Default Environment, the User Default Environment, and the Modified Print Environment (the Overlay Environment is described in Chapter 12, *Macros*).

Default settings refer to the settings programmed into the printer at the factory or settings selected using the control panel. The term "default" simply refers to the settings the printer uses unless printer commands select other settings.

Each time a job is printed, some of the printer's feature settings may be changed from their default values to produce the desired printed output for that job. After the job prints, the job-specific feature settings are longer be required, since the next job is likely to have different output requirements. The next job should clear all previous job settings by performing a reset. This allows a job to start with the default settings as a known base set, and vary only those settings that are needed. Starting with the default environment at the beginning of each print job eliminates the need to set every feature each time a job is run.

Note The print environment features presented in this chapter are for a variety of HP LaserJet printers. Not all of the features are applicable to every printer. To identify variations and default settings for specific printers, refer to Chapter 3 of the *PCL 5 Comparison Guide*.

Factory Default Environment

A factory default is a feature setting programmed into the printer at the factory. The group of all of the printer's feature settings set to their factory settings is referred to as the Factory Default Environment. These features are described in this manual. Since the HP-GL/2 features are used for HP-GL/2 operation only, the print environment features are separated, for convenience, into two lists or contexts: PCL and HP-GL/2. Table 3-1 lists typical PCL print environment features, and Table 3-2 lists typical HP-GL/2 print environment features.

Note he factory default settings for the printer features are not shown here, since they differ depending on the printer model. To identify variations and default settings for specific printers, refer to Chapter 3 of the *PCL 5 Comparison Guide*.

Table 3-1 Factory Default Print Environment Features – PCL Context

JOB CONTROL	**FONTS**[1]
Number of Copies[2]	Symbol Set[2]
Duplex[2]	Spacing
Binding[2, 3]	Pitch[4]
Registration	Height[5]
Tray[2]	Style
Manual Feed[2]	Stroke Weight
User-defined Units	Typeface
	Underlining Mode

1. The font characteristics are determined by the default font. The default font can be the factory default font or the user-selected default font from the printer's control panel or from a font cartridge containing a default font.
2. User default values may be selected by the user from the printer's control panel for these items.
3. Selectable from the printer control panel if duplex is selected.
4. Selectable from the printer's control panel if a fixed-space scalable font has been selected as the user default.
5. Selectable from the printer's control panel if a proportional scalable font has been selected as the user default.

Table 3-1 Factory Default Print Environment Features – PCL Context *(continued)*

PAGE CONTROL	FONT MANAGEMENT
Print Direction	Font ID
Orientation[2]	Character Code
Page Size[2]	Symbol Set ID
Paper Source	**RASTER GRAPHICS**
Vertical Motion Index[2]	Left Graphics Margin
Horizontal Motion Index[2]	Resolution
Top Margin	Compression Mode
Text Length	Raster Height
Left Margin	Raster Width
Right Margin	Raster Graphics Presentation Mode
Perforation Skip	**PICTURE FRAME**
Line Termination	Picture Frame Width
PRINT MODEL	Picture Frame Height
Current Pattern	Picture Frame Anchor Point
Source Transparency Mode	HP-GL/2 plot Horizontal Size
Pattern Transparency Mode	HP-GL/2 Plot Vertical Size
Pattern Reference Point	**MACRO**
Pattern Rotation	Macro ID

1. The font characteristics are determined by the default font. The default font can be the factory default font or the user-selected default font from the printer's control panel or from a font cartridge containing a default font.
2. User default values may be selected by the user from the printer's control panel for these items.
3. Selectable from the printer control panel if duplex is selected.
4. Selectable from the printer's control panel if a fixed-space scalable font has been selected as the user default.
5. Selectable from the printer's control panel if a proportional scalable font has been selected as the user default.

Table 3-1 Factory Default Print Environment Features – PCL Context *(continued)*

RECTANGULAR AREA FILL	TROUBLESHOOTING
Horizontal Rectangle Size	End-of-Line Wrap
Vertical Rectangle Size	Display Functions
Pattern (Area Fill) ID	
STATUS READBACK	
Current Location Type	
Current Location Unit	

1. The font characteristics are determined by the default font. The default font can be the factory default font or the user-selected default font from the printer's control panel or from a font cartridge containing a default font.
2. User default values may be selected by the user from the printer's control panel for these items.
3. Selectable from the printer control panel if duplex is selected.
4. Selectable from the printer's control panel if a fixed-space scalable font has been selected as the user default.
5. Selectable from the printer's control panel if a proportional scalable font has been selected as the user default.

Table 3-2 Factory Default Print Environment Features – HP-GL/2 Context

CHARACTER GROUP	POLYGON GROUP
Symbol Set	Polygon Buffer
Font Spacing	Polygon Mode
Pitch	**LINE AND FILL ATTRIBUTE GROUP**
Height	Line Type
Posture	Line Type Repeat Length
Stroke Weight	Line Cap
Typeface	Line Join
Character Direction	Miter Limit
Character Direction Mode	Pen Turret
Character Size Mode	Pen Width
Character Width	Pen Width Selection Mode
Character Height	Selected Pen
Character Slant	Symbol Mode
Extra Horizontal Space	Fill Type
Extra Vertical Space	User-defined Line Type
Character Fill Mode	Anchor Corner
Label Origin	User-defined Fill Types
Label Terminator	**PALETTE EXTENSION**
Transparent Data Mode	Number of Pens
Primary Font ID	Transparency Mode
Secondary Font ID	Screened Vector
Scalable or Bitmap Font	**CONFIGURATION & STATUS GROUP**
VECTOR GROUP	Scale Mode
Plotting Mode	Window
Pen State	Coordinate System Orientation
	P1, P2

User Default Environment

There are several PCL features in the printer for which user defaults may be selected from the printer's control panel. User default settings are stored in the User Default Environment and are retained even if the printer is turned OFF. Some of these features are listed below (for a complete set of the control panel user default features refer to Chapter 3 of the *PCL 5 Comparison Guide*.)

- Number of Copies
- Font Selection (Font Source, Font Number, Pitch[1] or Point Size[1])
- Duplex/Simplex
- Binding Edge[2]
- Tray
- Paper (Page Size: Paper and Envelopes)
- Manual Feed
- Orientation
- Symbol Set

Notes

Refer to "*Horizontal Motion Index (HMI) Command*" in Chapter 5 for the implications of setting the user default font (source, number, pitch or point size).

Refer to the printer *User's Manual* for instructions on how to select these user defaults from the control panel.

The PJL (Printer Job Language) "SET" command overrides the PCL user default environment for the duration of a PJL job. If PJL is not active, then the PCL user default environment has precedence. Refer to "PCL Commands" in Chapter 1 for more information.

1. For scalable typefaces: fixed-space typefaces are selected only by pitch; and proportionally-spaced typefaces are selected only by point size.
2. Selectable only if duplex is On.

Modified Print Environment

The current printer feature settings constitute the Modified Print Environment. Whenever a feature setting is altered using escape sequences, the new setting is recorded in the Modified Print Environment.

The Modified Print Environment is saved during a macro **call** or **overlay** and restored upon its completion.

A Modified Print Environment consists of the current settings for the items listed in Table 3-1 and Table 3-2, with the exception of the items listed in Table 3-3, below.

Table 3-3 Items Not Included in Modified Print Environment

PCL Context	HP-GL/2 Context
Overlay environment	HP-GL/2 overlay environment
Current cursor position	Duplex registration
Cursor position stack	Polygon buffer
Downloaded fonts/macros	
User-defined patterns	

Note Not all of the Modified Print Environment features are applicable to every printer. To identify variations and default settings for specific printers, refer to the *PCL 5 Comparison Guide*.

Resetting the Print Environment

Resets are used to return the printer to a known environment. Depending on the type of reset performed, the printer returns to either the User Default Environment or the Factory Default Environment.

Printer Reset

A Printer Reset restores the User Default Environment and deletes temporary fonts, macros, user-defined symbol sets, and patterns. A Printer Reset is performed by sending the $^{E}{}_{C}$**E** command, or through the printer's control panel (see the printer *User's Manual*). The Printer Reset command is described in Chapter 4, *PCL Job Control Commands*.

The $^{E}{}_{C}$**E** command prints any partial pages of data that may have been received. The control panel [RESET] discards any formatted pages which have not yet been printed.

Both resets ($^{E}{}_{C}$**E** and the control panel [RESET]) return the HP-GL/2 settings to their default values. $^{E}{}_{C}$**E** used in HP-GL/2 mode returns the printer to PCL mode in addition to resetting the print environment. The HP-GL/2 **IN (Initialize)** command resets HP-GL/2 settings to their default values without affecting the PCL settings (refer to the Initialize command described in Chapter 19, *The Configuration and Status Group*, for additional information).

Notes

Hewlett-Packard strongly recommends the use of both the $^{E}{}_{C}$**E** command and the $^{E}{}_{C}$**%–12345X** command (Universal Exit Language/Start of PJL — also referred to as the **UEL Command**) at the beginning and end of each job. (The order of these commands is critical. Refer to Table 4-1 for an example of their usage.)

The UEL Command ($^{E}{}_{C}$**%–12345X**) has the same effect as the $^{E}{}_{C}$**E** command, and also enters PJL Mode of operation for printers that support PJL (refer to "Universal Exit Language Command" in Chapter 4 for more information). The $^{E}{}_{C}$**E** command should be included to ensure backward compatibility (the UEL command is ignored if received by a printer that does not support PJL).

Cold Reset

A Cold Reset restores the Factory Default Environment which includes resetting the control panel items to their factory default settings. A Cold Reset is performed by power cycling the printer while holding [ON LINE] until a **08 COLD RESET** is displayed.

4 PCL Job Control Commands

Introduction

A job typically consists of three parts:

- Commands providing job control.
- Commands providing page control.
- Print data.

Table 4-1 Structure of a Typical Job

EC%–12345X	UEL Command (exit language)
ECE	Printer Reset Command.
Preamble	Job Control Commands.
Page 1	Page Control Commands. Data
Page 2	Page Control Commands.[1] Data.
• • •	• • •
Page n	Page Control Commands. Data.
ECE	Printer Reset Command.
EC%–12345X	UEL Command (exit language).

1. If a number of consecutive pages within a job have the same format (such as margins, VMI, HMI, etc.), the associated page control commands only need to be sent once for that group of pages.

This chapter describes the commands providing job control. Job control commands are usually grouped together and sent at the beginning of a job. Page control commands and data are associated with each printed page of a job. Job control commands include the following:

- Printer Reset.
- Universal End of Language/Start of PJL.
- Number of Copies.
- Simplex/Duplex Print.
- Left and Top Offset Registration.
- Duplex Page Side Selection.
- Job Separation.
- Output Bin Selection.
- Unit of Measure.

Printer Reset Command

Receipt of the Printer Reset command restores the User Default Environment, deletes temporary fonts, macros, user-defined symbol sets and patterns. It also prints any partial pages of data which may have been received.

EC **E**

Notes

Hewlett-Packard strongly recommends the use of both the EC E command and the EC**%–12345X** command (Universal Exit Language/Start of PJL — also referred to as the **UEL Command**) at the beginning and end of each job. (The order of these commands is critical. Refer to Table 4-1 for an example.)

The UEL Command (EC**%–12345X**) has the same effect as the EC E command, and also enters PJL Mode of operation for printers that support PJL (refer to the next section, "“Universal Exit Language Command” for more information). The EC E command should be included to ensure backward compatibility (the UEL command is ignored if received by a printer that does not support PJL).

Universal Exit Language Command

The Universal Exit Language (**UEL**) command causes the PCL printer language to shut down and exit. Control is then returned to the Printer Job Language (PJL). Both PCL 5 and HP-GL/2 recognize this command.

^{E}C **% – 1 2 3 4 5 X**

Default = N/A

Range = –12345

This command performs the following actions:

- Prints all data received before the Exit Language command.
- Performs a printer reset (same effect as ^{E}C E).
- Shuts down the PCL 5 printer language processor.
- Turns control over to PJL.

Notes

Hewlett-Packard strongly recommends the use of both ^{E}C E (printer reset) and ^{E}C**%–12345X** (UEL command) at the beginning and end of each job. (The order of these commands is critical. Refer to Structure of a Typical JobTable 4-1 for an example.)

The UEL Command (^{E}C**%–12345X**) has the same effect as the ^{E}C E command, and also enters PJL Mode of operation for printers that support PJL. The ^{E}C E command should be included to ensure backward compatibility (the UEL command is ignored if received by a printer that does not support PJL).

Number of Copies Command

The Number of Copies command designates the number of printed copies of each page.

EC & ℓ # X

\# = Number of copies (1 to 32767 maximum)

Default = 1 (Configurable from control panel)

Range = 1-32767
(Values 32767 execute as 32767 values 1 are ignored. Maximum number of copies=99 for LaserJet II, IIP, III, IIID, IIIP and earlier LaserJet printers.)

This command can be received anywhere within a page and affects the current page as well as subsequent pages.

Example

To print 3 copies of a page, send:

EC&ℓ3X

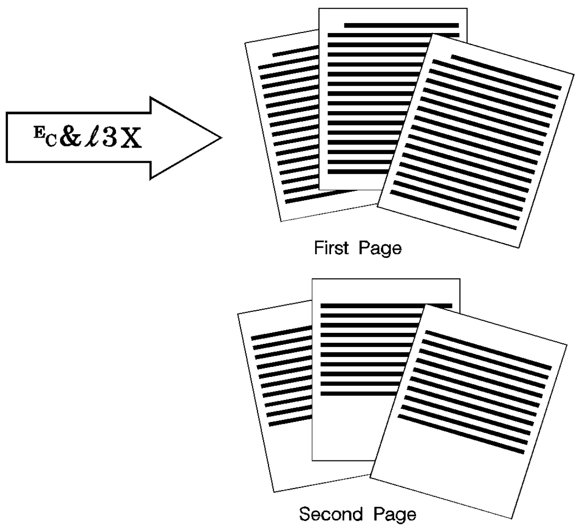


Figure 4-1 Number of Copies

Note

The HP-GL/2 Replot (RP) command is inactive for PCL 5 printers; use the Number of Copies command for multiple HP-GL/2 plots. To be effective, the Number of Copies command must be issued from PCL prior to closing the page on which the plot is defined.

Simplex/Duplex Print Command

This command designates either simplex or duplex printing mode for duplex printers. Simplex mode prints an image on only one side of a sheet (page). Duplex mode prints images on both sides of a sheet.

^{E}C & ℓ # S

\#   =   0 - Simplex
1 - Duplex, Long-Edge Binding
2 - Duplex, Short-Edge Binding

Default **=** 0

Range **=** 0-2 (Other values ignored)

Long-Edge bound duplexed pages are bound along the length of the physical page (see Figure 4-2). Short-edge bound duplexed pages are bound along the width of the physical page (see Figure 4-3).

Selecting long-edge binding usually results in font rotation. This may be a concern if available user memory is critical.

Note If this command is received by a printer which does not contain the duplex feature, it is ignored. Printers which do not contain the duplex feature print in simplex mode (front side of sheet) only.

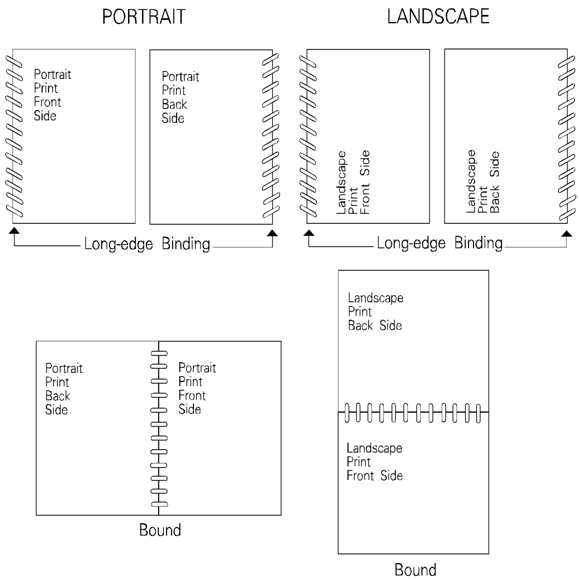


Figure 4-2 Long-Edge Binding Mode

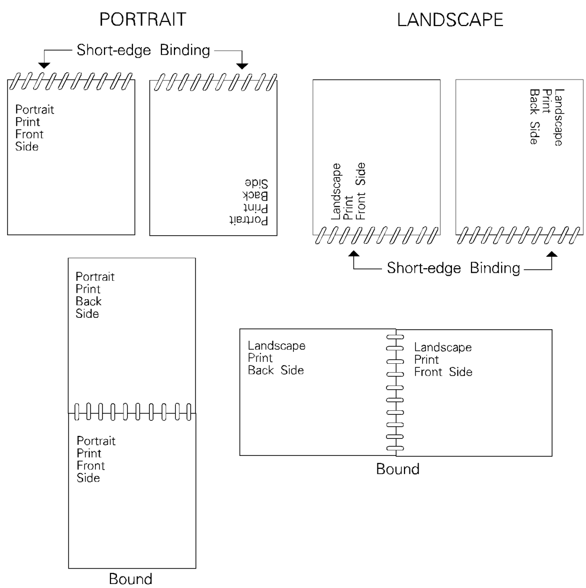


Figure 4-3 Short-Edge Binding Mode

Left Offset Registration Command

The Left (long-edge) Offset Registration command designates the position of the logical page across the width (short side) of the physical page. This command can be used to adjust the text position on the page to allow additional room for the page binding.

^{E}C **& ℓ # U**

\# = The number of decipoints (1/720 inch)

Default = 0

Range = –32767 to 32767

The value (#) is a signed number valid to 2 decimal places. The units are decipoints. Positive values cause the logical page, regardless of orientation, to move right along the width of the physical page, except on the back side (duplex print) of sheets printed in long-edge binding duplex mode, where positive values cause it to move left (refer to Figure 4-4 and Figure 4-5).

Negative values cause the logical page, regardless of orientation, to move left along the width of the physical page, except on the back side of sheets printed in long-edge binding duplex mode, where negative values cause it to move right (refer to Figure 4-4 and Figure 4-5).

Notes

The +/– value is absolute with respect to the default position of the logical page along the width of the physical page. It is not relative to the present location.

The registration commands may cause data loss by moving the logical page outside the printable area.

This command has the same effect regardless of orientation.

This command can be used in both simplex and duplex modes.

Top Offset Registration Command

The Top (short-edge) Offset Registration command designates the position of the logical page along the length (long side) of the physical page.

^{E}C **& ℓ # Z**

\# = The number of decipoints (1/720 inch)

Default = 0

Range = –32767 to 32767

The value (#) is a signed number valid to 2 decimal places. The units are decipoints. Positive values cause the logical page, regardless of orientation, to move down along the length of the physical page, except on the backside of sheets printed in short-edge binding duplex mode, where positive values cause it to move up (refer to Figure 4-4 and Figure 4-5).

Negative values cause the logical page, regardless of orientation, to move up, along the length of the physical page, except on the backside of sheets printed in short-edge binding duplex mode, where negative values cause it to move down (refer to Figure 4-4 and Figure 4-5).

Notes

The +/– value is absolute with respect to the default position of the logical page along the length of the physical page. It is not relative to the current location of the logical page.

The registration command may cause data loss by moving the logical page outside the printable area.

This command has the same effect regardless of orientation.

This command can be used in both simplex and duplex modes.

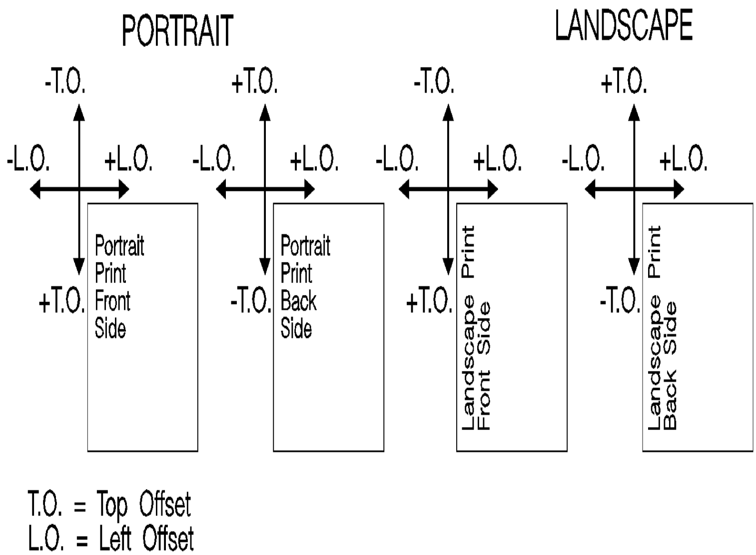


Figure 4-4 Short-Edge Binding Mode Offsets

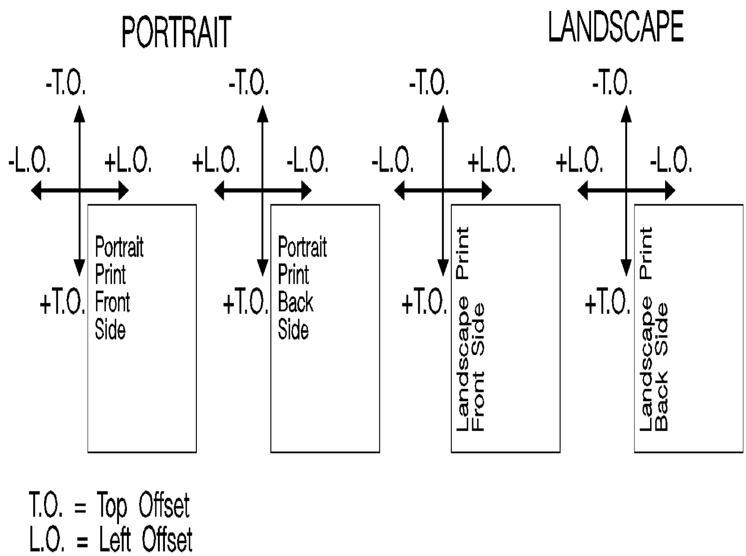


Figure 4-5 Long-Edge Binding Mode Offsets

Duplex Page Side Selection Command

The Duplex Page Side Selection command causes a Form Feed and designates which side of the sheet to print.

The ability to skip a page while duplexing may be required at certain locations in a document. For example, a chapter typically begins on the front side of a page.

$^{E}{}_{C}$**&a#G**

\# = 0 - Select next side
1 - Select front side
2 - Select back side

Default = 0

Range = 0-2 (All other values ignored)

If this command is received by a printer which does not have duplex or if duplexing is not enabled, these commands just eject the current page (sheet), positioning the cursor at the default position on the next page.

Example

To print on the front side of a page, regardless of the current side, send the following:

$^{E}{}_{C}$**&a1G**

In this example, if the printer is currently formatting a front side, it will stop formatting, eject that page (sheet, skipping the back side), and begin printing on the next front page.

Job Separation Command

Job separation provides a means of identifying one print job from others in the printer's output tray. It usually does this by physically offsetting one print job from the next.

The Job Separation command toggles the printer's separation mechanism. This command must be sent between each job to enable the separation mechanism.

^{E}C **& ℓ 1 T**

HP recommends that the Job Separation command be included at the end of each job, just before the Printer Reset command. HP also recommends that this command be included in the programs even though printers with job separation are not currently being used. This ensures that if a printer with job separation is eventually added, job separation will be performed.

If this command is received by a printer which does not have job separation, the command is ignored.

Note It is possible to perform job offset in printers which do not have a mechanical offset mechanism but have dual paper trays. In dual bin printers, job offset can be performed by placing colored paper in the second tray and using Paper Source command to select the tray to feed a blank sheet of colored paper at the end of a job. This method should only be used in special cases where the end user can control its use, for example, the program should only be available for a dual bin printer which can always contain colored paper in one tray.

Output Bin Selection Command

The **Output Bin Selection** command selects either of the two output paper bins (upper or lower [rear]) for paper output.

^{E}C **& ℓ # G**

\# = 1 - Upper Output Bin
2 - Lower (Rear) Output Bin

Default = Upper Output Bin

Range = 1, 2

Note If this command is received by a printer which does not contain the dual output bin feature, it is ignored.

Unit of Measure Command

The Unit of Measure command establishes the unit of measure for **PCL Unit** cursor movements.

E_C **& u # D**

\# =Number of units-per-inch

Default = 300

Range = 96, 100, 120, 144, 150, 160, 180, 200, 225, 240, 288, 300, 360, 400, 450, 480, 600, 720, 800, 900, 1200, 1440, 1800, 2400, 3600, 7200.

The value field defines the number of units-per-inch used in the following commands:

- Vertical Cursor Position (PCL Units).
- Horizontal Cursor Position (PCL Units).
- Vertical Rectangle Size (PCL Units).
- Horizontal Rectangle Size (PCL Units).

In addition, the current unit of measure setting affects the HMI setting, which in turn determines how cursor movement values are rounded. This affects the result of the following commands:

- Horizontal Cursor Position (Columns).
- Horizontal Tab (HT control code).
- Space (SP control code).
- Backspace (BS control code).
- Bitmap Character Delta X ("Delta X (SI)," Chapter 11).

For example, if the unit of measure is set to 96 (one PCL Unit = 1/96 inch), then the HMI is rounded to the nearest 1/96 inch. If the unit of measure is set to 300 (one PCL Unit = 1/300 inch), the HMI is rounded to the nearest 1/300 inch.

Note HMI is set either as a result of font selection or through the use of the HMI command. The rounding behavior just described only applies when the HMI is at its default setting (derived from the currently selected font). If the HMI Command was used to override the HMI setting, the rounding behavior described above does not apply. (See "Horizontal Motion Index (HMI) Command" in Chapter 5 for more information.)

The current unit of measure setting affects all PCL Unit moves, horizontal and vertical rectangle size, bitmap and scalable font metrics (how the cursor moves after printing a character). The Unit of Measure command does not affect the interpretation of binary raster data (bitmap fonts, raster graphics or patterns).

Once the units of measure is changed, it stays in effect until another is selected or the printer is reset. A control panel or ^{E}C **E** reset returns the current unit of measure setting back to the device default setting (300).

The units value is part of the modified print environment. As such, it is saved and restored whenever a macro is called or an overlay invoked, and defaulted when establishing the overlay environment in preparation for an overlay.

Note Values out of range are mapped to the supported value with the minimum relative error. For example, a unit selection of 4801 would be mapped to 7200, since the relative error (0.3332) is less than the relative error when mapped to 3600 (0.3336):

$$\frac{|14801 - 7200|}{7200} = 0.3332 < \frac{|4801 - 3600|}{3600} = 0.3336$$

{|4801-7200| over 7200}=0.3332 < {| 4801-3600|over 3600}=0.3336

Figure 4-6 compares a 4-unit vertical and horizontal cursor move with a unit of measure setting of 100 versus 200 units-per-inch. Note that the cursor move distance is halved when the Units per inch is doubled.

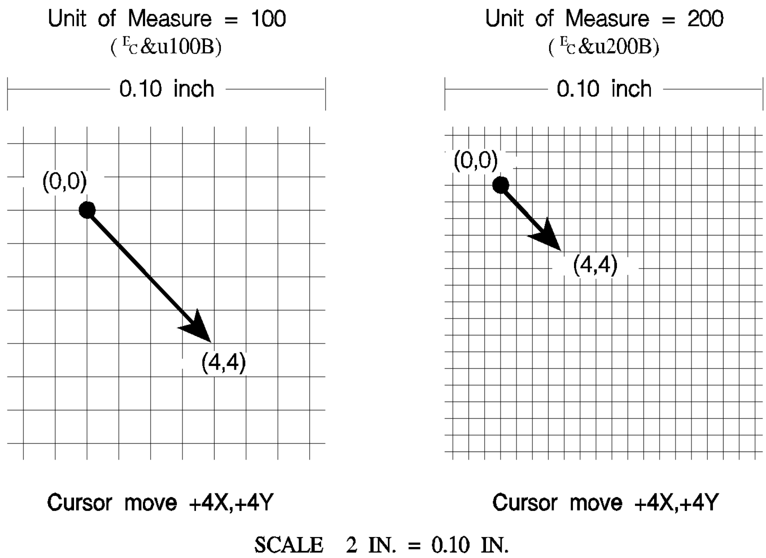


Figure 4-6 Cursor Moves at Different Unit of Measure Settings

The printer's physical **dot size** has **no direct bearing** on the size of **PCL Units** used in cursor movements. In addition, PCL Units are **not affected by the current control panel or PJL resolution setting.**

Note

If no other unit of measure value has been specified, then the default is one Unit equals 1/300 inch. In this case, a cursor movement of 450 Units moves the cursor 1.5 inches, whether printed at 300 or 600 dpi print resolution.

5 Page Control Commands

Introduction

Page control commands and data are associated with each printed page of a job. These commands determine such features as page source, size, orientation, margins, and text spacing. This chapter describes the commands providing page format control.

Note If a number of consecutive pages within a job have the same format, the associated page control commands should be sent only once for that group of pages. Remember, once a PCL command is set, it remains in effect until changed by another command.

Page Size Command

The Page Size command designates the size of the paper which in turn defines the size of the logical page.

^{E}C **& ℓ # A**

PAPER:

\# = 1 - Executive (7¼ x 10½ in.)
2 - Letter (8½ x 11 in.)
3 - Legal (8½ x 14 in.)
6 - Ledger (11 x 17 in.)
26 - A4 (210mm x 297mm)
27 - A3 (297mm x 420mm)

ENVELOPES:

\# = 80 - Monarch (Letter - 3 7/8 x 7½ in.)
81 - Com-10 (Business - 4 1/8 x 9½ in.)
90 - International DL (110mm x 220mm)
91 - International C5 (162mm x 229mm)
100 - International B5 (176mm x 250mm)

Default = 2 (Configurable from Control Panel)

Range = 1, 2, 3, 6, 26, 27, 80, 81, 90, 91, 100 (Other values ignored)

Note Refer to the "PCL Feature Support Matrix" in Chapter 1 of the *PCL 5 Comparison Guide* or the printer *User's Manual* for lists of supported paper and/or envelope sizes on specific printers.

Upon receipt of this command any unprinted pages are printed, the top margin, text length, and left and right margins are set to their user defaults, and any automatic macro overlay is disabled. The cursor is moved to the left edge of the logical page at the top margin on the following page (see Figure 5-5). Also, certain HP-GL/2 state variables are reset (refer to Table 5-1, under "Sending a Page Size Command:").

The factory default Page Size is **Letter** (A4 for 220v option printer); however, a user default Page Size may be selected from the control panel. The Page Size command takes precedence over the printer's control panel FORM setting.

If the Page Size command selection differs from that of the installed paper tray size and the requested page size is not currently available in another non-secure source, then a message is displayed on the control panel requesting installation of a paper tray of the specified size.

Note The Page Size command is not supported on the **HP LaserJet**,, **LaserJet+**, or the **LaserJet 500**+ **printers**.

Example

To select a legal size page, send:

^{E}C **& ℓ3A**

If the current source paper tray is “LETTER” and the requested page size is not currently available in another non-secure source the following attendance message is displayed:

xx [1] “LOAD LEGAL”

1 *xx* = A two-letter mnemonic depending on the printer.

When the printer senses the existing paper supply has been removed and replaced with the requested paper size, the print job continues automatically.

PAPER SOURCE COMMAND

The Paper Source command designates the location to feed paper, or it prints the current page.

EC & ℓ # H

\# = 0 - Print the current page
(paper source remains unchanged).
1 - Feed paper from the a printer-specific tray.
2 - Feed paper from manual input.
3 - Feed envelope from manual input.
4 - Feed paper from lower tray.
5 - Feed from optional paper source.
6 - Feed envelope from optional envelope. feeder[1]

1 Must be used in conjunction with Page Size command, envelope selection.

Default = Printer Dependent (Configurable from Control Panel)

Range = Printer Dependent

Note

Not all HP LaserJet printers support all possible paper sources. The implementation of paper source locations varies slightly from printer to printer. Refer to the "PCL Feature Support Matrix" in Chapter 1 of the *PCL 5 Comparison Guide* or the printer *User's Manual* for paper source feature implementation details for specific HP LaserJet printers.

The Paper Source command causes the current page to be printed and the cursor to be moved to the left edge of the logical page at the top margin position for the next page (see Figure 5-5).

Example

To feed paper from the manual feed slot, send:

EC&ℓ2H

If the selection requires operator action (such as manually feeding paper), a printer message appears in the display, prompting for the appropriate action (see the printer *User's Manual* for specific behavior).

Logical Page Orientation Command

Orientation defines the position of the logical page and the default direction of print with respect to the physical page as shown in Figure 5-1.

^{E}C & ℓ # O

\# = 0 - Portrait
1 - Landscape
2 - Reverse Portrait
3 - Reverse Landscape

Default = 0

Range = 0-3 (Other values ignored)

Notes

This command can be used only once per page. To print multiple directions per page use the Print Direction command.

This command affects the HP-GL/2 environment (refer to Table 5-1 and the ***“HP-GL/2 and PCL Orientation Interaction” section in Chapter 15 for additional information).

The Orientation command causes the page length, top margin, text length, left and right margins, horizontal motion index (HMI), and vertical motion index (VMI) to return to their user default values, and disables the automatic macro overlay. All data received prior to this command is printed, and a Form Feed and Carriage Return executed. The cursor is moved to the left edge of the logical page at the top margin cursor position (see Figure 5-5).

The factory default orientation is portrait. Landscape orientation may be selected as the user default orientation using the control panel.

Note

The HP LaserJet 2000, LaserJet IID, LaserJet IIP and all PCL 5 printers automatically rotate all fonts to the current orientation.

Table 5-1 shows how changing certain PCL features (such as a reset, orientation, page size or page length) or changing the HP-GL/2 picture presentation directives (picture frame width or height, horizontal or vertical plot size, or picture frame anchor point) affect the HP-GL/2 state variables.

Table 5-1 HP-GL/2 State Variables

EcE **or** **Control Panel Reset:**	• executes "IN" command • defaults picture frame • defaults picture frame anchor point • defaults HP-GL/2 plot size
Changing Orientation **or** **Sending a Page Size Command:**	• Defaults picture frame anchor point. • Defaults picture frame. • Defaults HP-GL/2 plot size. • Defaults P1 and P2 ("IP;"). • Defaults soft-clip window ("IW;'"). • Clears the polygon buffer ("PM0;PM2"). • Updates the current position to the lower-left corner of the picture frame (P1).
Redefinition of the horizontal and/or vertical picture frame:	• Defaults P1 and P2 ("IP"). • Defaults soft-clip window ("IW"). • Clears the polygon buffer ("PM0;PM2"). • Updates the current position to the lower-left corner of the picture frame (P1).
Redefinition of the Picture Frame Anchor Point:	• Defaults P1 and P2 ("IP"). • Defaults soft-clip window ("IW"). • Clears the polygon buffer ("PM0;PM2"). • Updates the current position to the lower-left corner of the picture frame (P1).
Specifications of a New HP-GL/2 Plot Size:	• Changes the picture frame scaling factor.

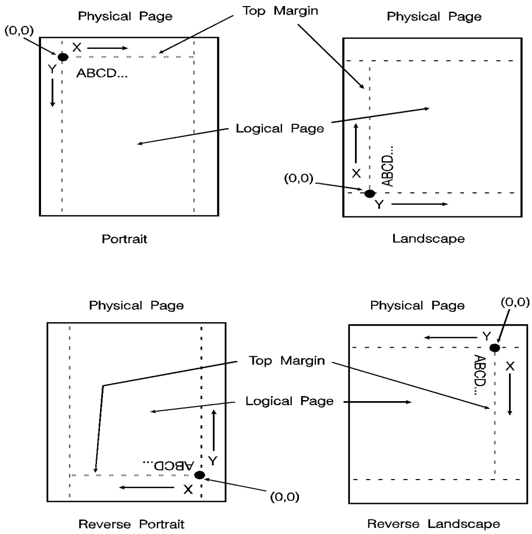


Figure 5-1 Page Orientation With Default Print Direction

The orientation of the HP-GL/2 picture is also affected by the logical page orientation. Figure 5-2 illustrates the effect of logical page orientation on the HP-GL/2 picture orientation. It is possible to alter the HP-GL/2 picture orientation within the logical page using the HP-GL/2 RO command (refer to Chapter 21, for additional information).

Most HP-GL/2 state variables retain their previous HP-GL/2 value upon receipt of this command (they are not affected by PCL mode). However, certain changes to the PCL state can affect the HP-GL/2 state (see Table 5-1).

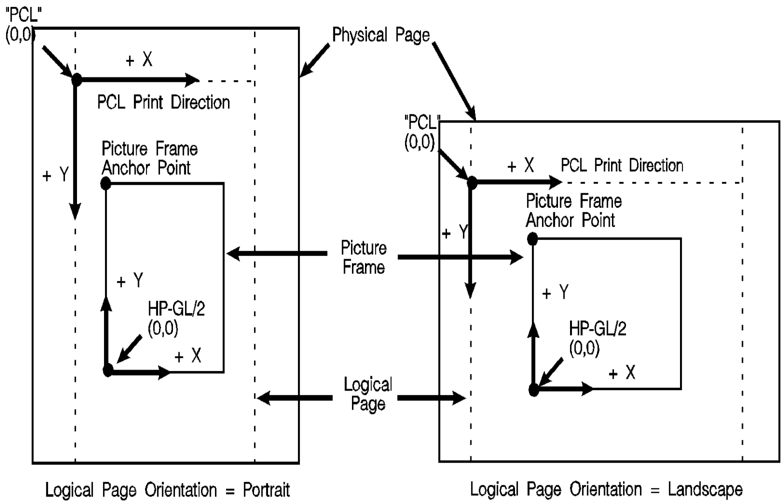


Figure 5-2 HP-GL/2 Picture Orientation with Respect to Logical Page Orientation

Print Direction Command

The Print Direction command rotates the logical page coordinate system **with respect to the current orientation** without performing a page eject. This rotation is performed in 90° increments in a **counterclockwise** direction. This allows printing in four directions on the same page.

EC & a # P

#	=	0 - 0° rotation.
		90 - 90° ccw rotation.
		180 - 180° ccw rotation.
		270 - 270° ccw rotation.

Default = 0

Range = 0, 90, 180, 270 (Other values ignored)

Changing the print direction causes the following:

- The print origin moves with the logical page rotation. For example, rotating a default page (portrait orientation, 0° print direction) 90° causes data to print in the landscape direction across the "portrait" page.
- The margins are translated (when the print direction changes by 90°, the left margin becomes the new top margin, the former top margin becomes the new right margin, etc.)
- The cursor position remains at the same physical location.
- All subsequent printing (characters, area fill patterns, raster images) is rotated to coincide with the new print direction.
- Any current raster graphics end when the print direction changes.
- Print Direction does not default HMI.

Note

The Print Direction Command does **not** affect HP-GL/2 vector graphic images. HP-GL/2 graphics can be rotated only with the Orientation command (EC**&ℓ#O**) or the HP-GL/2 "RO" command.

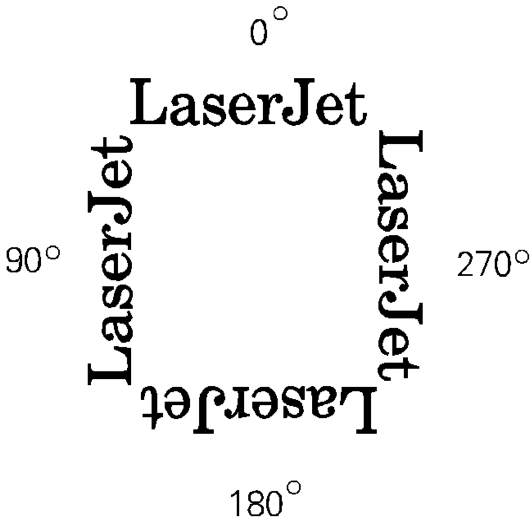


Figure 5-3 Changing Print Direction on a Page

Text Area

Text printing may be restricted to a specific area within the logical page using the Left Margin, Right Margin, Top Margin, Text Length, and Perforation Skip Mode commands. This area is known as the text area.

The left margin defines the distance between the left edge of the logical page and the left edge of the text area. The right margin defines the distance between the left edge of the logical page and the right edge of the text area. The width of the text area is the distance between the left and right margins. The top margin defines the distance between the top of the logical page and the top of the text area. The text length defines the length of the text area which in effect defines the bottom margin. The perforation region is the distance from the bottom of the text area to the top of the text area (top margin) on the next page. The text area is shown in Figure 5-4.

In general, characters are printed when they fall within the text area. However, characters can be printed between the bottom of the text area and the top of the text area on the next page only if perforation skip is disabled. Characters are printed outside the text area if a cursor move escape sequence positions the cursor outside the text area (but within the printable area). Characters that fall on (or outside) a margin as a result of printing a character string, are clipped (not printed).

Notes

Attempting to print characters across a margin results in the characters being discarded.

The default text area and the default HP-GL/2 picture frame are the same.

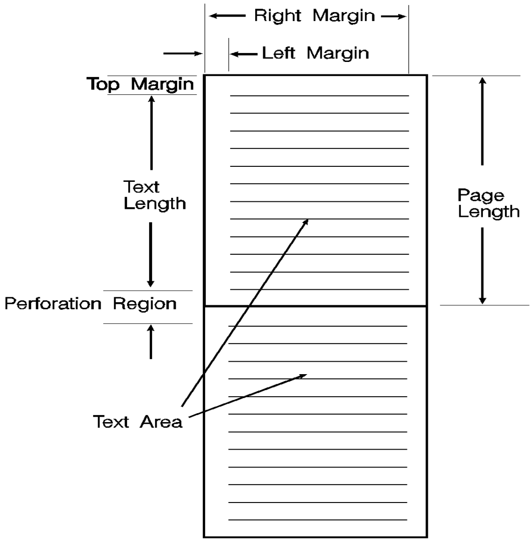


Figure 5-4 Text Area Within the Page

Left Margin Command

The Left Margin command sets the left margin to the left edge of the specified column.

^{E}C **& a # L**

= Column number

Default = Column 0 (Left bound of logical page)

Range = 0 - Right margin

The first column within a line is column 0, which is located at the left edge of the logical page (the HMI setting defines the distance between columns, which thereby defines the maximum number of columns on the logical page). If the value field specifies a column greater than the current right margin, the command is ignored.

Margins represent a physical position and once set do not change with subsequent changes in HMI.

If the cursor is to the left of the new left margin, the cursor is moved to the new left margin.

Example

To set the left margin to column 5, send:

^{E}C**&a5L**

Right Margin Command

The Right Margin command sets the right margin to the right edge of the specified column.

^{E}C **& a # M**

\# = Column number

Default = Logical Page right bound

Range = Current left margin - Logical page right bound

The maximum right column is located at the right edge of the logical page (the HMI setting defines the distance between columns, which thereby defines the maximum number of columns on the logical page). If the value field specifies a column which is greater than the right edge of the logical page, the right margin is set to the right edge of the logical page. If the value field specifies a column less than the left margin, the command is ignored.

Margins represent a physical position and once set do not change with subsequent changes in HMI.

If the cursor position is to the right of the new right margin, the cursor is moved to the new right margin.

Example

To set the right margin to column 45, send:

^{E}C**&a45M**

Clear Horizontal Margins Command

The Clear Horizontal Margins command resets the left and right margins. The left margin is set to the left edge of the logical page (column 0) and the right margin is set to the right edge of the logical page.

^{E}C **9**

Top Margin Command

The Top Margin command designates the number of lines between the top of the logical page and the top of the text area.

EC & ℓ # E

= Number of lines

Default = 1/2 inch down from top of logical page[1]

Range = 0 - Length of logical page (Other values ignored)

1 If logical page length is <½inch, then the top margin is set to top of logical page.

The Top Margin command is ignored if the value field (#) is greater than the current logical page length or if the current VMI is 0 (VMI defines the distance between lines of text).

Receipt of a Top Margin command resets the text length according to the following equation:

Text Length = (logical page length in inches) - (top margin in inches + 1/2 inch)

The top margin represents a physical position and once set does not change with subsequent changes in VMI or line spacing.

The vertical cursor position for the first line of print is determined by the current values of the top margin and VMI using the following equation:

first line in inches = top margin in inches + (0.75 X VMI)

Note The default cursor position is not located at the intersection of the top margin and the left bound of the logical page (refer to Figure 5-5). The cursor is actually positioned down 75% of the VMI distance (0.75 × VMI) from the top margin. This positions the cursor at the relative base line position of a character cell for correct character positioning.

Example

To set the top margin to line 4, send:

EC&ℓ4E

Note The first line of the logical page is line 0.

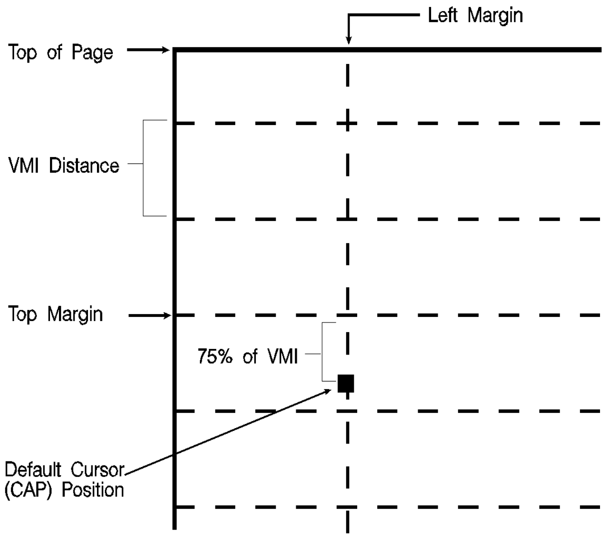


Figure 5-5 Margin Cursor Positioning

Text Length Command

The Text Length command designates the number of lines (at a given VMI) within the logical page available for printing text, the text area. This effectively defines the bottom margin.

^{E}C **& ℓ # F**

\# = Number of lines

Default = 1/2 inch less than maximum text length1

Range = Logical page length minus top margin

1 Maximum text length = INT(logical page length - top margin). However, if the max text length is less than ½inch, the text length is set to the maximum allowable.

The value field (#) sets the text length in lines referenced from the top margin. If a value greater than the logical page length minus the top margin is specified or if the current VMI is 0, the command is ignored. The user default text length is invoked whenever the orientation, page length, page size, or top margin is changed. The user default text length is computed as follows:

$$\text{Integer portion of Text Length in Lines} = \left(\text{(logical page length in inches)} - \text{(top margin in inches)} - \text{(1/2 inch)} \times \frac{48}{\text{VMI}}\right)$$

$$\text{Factory Default Text Length in Lines} = \left(\text{(Logical page length in inches} - 1\text{ inch)} \times \frac{48}{8}\right)$$

Note The user default VMI is selectable using the control panel; VMI is calculated from the FORM menu setting.

Example

To select a text length of 60 lines, send:

^{E}C**&ℓ6ØF**

Perforation Skip Command

The perforation region is the distance from the bottom of the text area of one page to the top of the text area (top margin) of the next page. When perforation skip is enabled, a Line Feed or Half-Line Feed, which would move the cursor beyond the bottom of the text area, causes the cursor to move to the top of the text area on the next page. When perforation skip is disabled, a Line Feed or Half-Line Feed allows the cursor to move to the next line or half-line in the perforation region, allowing printing to continue there.

^{E}C **& ℓ # L**

\# = 0 - Disable
1 - Enable

Default = 1

Range = 0-1 (Other values ignored)

Whenever the perforation skip mode is changed, the top margin and page length are returned to their default values.

Note When perforation skip is disabled, some print lines can fall outside the printable area and be lost. If lines of data could fall into the unprintable area, perforation skip should be enabled.

Horizontal Motion Index (HMI) Command

The Horizontal Motion Index (HMI) command designates the width of the columns.

^{E}C **& k # H**

\# = Number of 1/120 inch increments.

Default **=** Determined by the pitch value in the default font header.

Range **=** 0 - 32767 (valid to four decimal places)

The value field is valid to 4 decimal places. A value of zero (0) indicates no horizontal motion.

When fixed pitch fonts are selected, all printable characters including the Space and Backspace characters are affected by HMI. When proportional fonts are selected, the HMI affects only the Space control code character.

HMI is reset to match the new font when any of the font characteristics are changed and when switching between primary and secondary fonts with Shift In and Shift Out.

HMI is equal to the pitch value in the font header. The factory default font's HMI is 12 (12/120 = 1/10 inch per character, or 10 characters per inch).

Note

When HMI is not specifically set using the HMI command, PCL cursor moves are rounded to the nearest full increment determined by the current unit of measure setting. For example, if the unit of measure is set to 96 (one PCL Unit = 1/96 inch), then the HMI is rounded to the nearest 1/96 inch. If the unit of measure is set to 300 (one PCL Unit = 1/300 inch), the HMI is rounded to the nearest 1/300 inch.

Example

To print the printer's resident 16.66 pitch Line Printer font at 17.75 cpi, send ^{E}C**(s16.66H** to select the Line Printer font, then send the command ^{E}C**&k6.76H** to change HMI. This value field is calculated as follows:

$$\text{Desired HMI} = \frac{\text{HMI units}}{\text{Desired CPI}} = \frac{120}{17.75} = 6.76 \text{ HMI}$$

Each character then occupies 6.76/120 inch or 1/17.75 inch.

To use Courier 12 point (10 cpi) and print 80 characters across A4 paper, requires adjusting the HMI value. The HMI value is calculated as follows:

$$\text{A4 Width (inches)} = \frac{\text{2338 dots wide*}}{\text{300 dots/in.}} = 7.793 \text{ inches}$$

* This value was obtained from Figure 2-3 which identifies the page sizes (in 300 dpi dots).

$$\text{\# char./inch} = \frac{\text{80 characters}}{\text{7.793 in.}} = 10.266 \text{ cpi}$$

$$\text{Desired HMI} = \frac{\text{120 HMI units}}{\text{10.266 char./inch}} = 11.689 \text{ HMI value}$$

Vertical Motion Index (VMI) Command

The Vertical Motion Index (VMI) command designates the height of the rows. (The vertical distance the cursor moves for a Line Feed operation.)

^{E}C **& ℓ # C**

= number of 1/48 inch increments between rows.

Default = 8

Range = 0 - Current logical page length up to a maximum of 32767

If the specified VMI is greater than the current logical page length, the command is ignored.

The value field is valid to 4 decimal places. A Ø in the value field indicates no vertical movement.

This command affects the Line Feed and Half-Line Feed spacing.

The factory default VMI is 8, which corresponds to 6 lines-per-inch. A user default VMI can be selected from the control panel using the FORM menu item (refer to the printer *User's Manual* for additional information).

Example

To designate a VMI of 6 (8 lines-per-inch) send:

^{E}C**&ℓ6C** (6/48 = 1/8 inch/line)

The following equation converts lines-per-inch spacing to VMI:

$$\text{VMI} = 48 \times \left(\frac{1}{\text{\# of desired lines per inch}} \right)$$

Note

A change in the control panel FORM setting results in a modification of VMI. If the Page Length command (^{E}C**&ℓ#P**) follows a VMI change, the physical size of the page is recalculated. Therefore, depending on the VMI modification made, the printer may request a different paper size.

Common VMI Settings

To print 66 lines per page on letter-size paper, in portrait orientation (with one-half inch top and bottom margins) send:

^{E}C**&ℓ7.27C**   7.27 = (10/66) x 48

To print 66 lines per page on letter or legal-size paper, in landscape orientation (with one-half inch top and bottom margins) send:

^{E}C**&ℓ5.45C**   5.45 = (7.5/66) x 48

Line Spacing Command

The Line Spacing command sets the number of lines printed per inch. Only the values listed below are valid.

EC & ℓ # D

#	=	1 - 1 lpi
		2 - 2 lpi
		3 - 3 lpi
		4 - 4 lpi
		6 - 6 lpi
		8 - 8 lpi
		12 - 12 lpi
		16 - 16 lpi
		24 - 24 lpi
		48 - 48 lpi

Default **=** 6

Range **=** 0,1,2,3,4,6,8,12,16,24,48 (Other values are ignored)

This command performs the same function as the Vertical Motion Index (VMI) command except that it identifies the VMI in lines-per-inch (lpi).

The factory default lines-per-inch setting is 6. A user default line spacing can be selected from the control panel using the FORM menu item.

Example

To select 12 lpi, send:

EC&ℓ12D

Note Once a PCL command sets a parameter, that parameter remains in effect until another command changes it. The most recently received command has precedence.

6 Cursor Positioning

Introduction

This section describes the cursor positioning commands.

Although the printer does not actually have a cursor, the PCL cursor position refers to the **Current Active Position (CAP)**, like the blinking underline character (cursor) used on most computers. This “cursor” identifies the current position on the page; the pointer, where a printing command begins laying out page data. The cursor can be moved anywhere within the logical page using a combination of horizontal and vertical cursor positioning commands and control codes.

In addition to cursor commands positioning the cursor, the cursor is automatically positioned after certain operations, such as printing characters and graphics. After printing a character, the cursor is positioned to the right, at a distance equal to the width of that character. This is controlled by the character design described under “Character Width” in Chapter 10, and allows printing characters without requiring a cursor position command for each character printed. When printing graphics, the cursor can also be positioned at a new location. These new positions are identified in the graphics sections.

HP-GL/2 vector graphics has its own HP-GL/2 cursor (referred to as the “pen”) that can be positioned within the HP-GL/2 addressable area. For additional information on HP-GL/2 pen positioning refer to Chapter 17, *An Introduction to HP-GL/2 Vector Graphics*.

Absolute vs. Relative Cursor Positioning

Either absolute or relative motion can be specified.

Absolute motion always specifies the distance to move referenced from the top margin at the left bound of the logical page (0,0), *regardless of the current active position* (CAP) (see Figure 6-1). An unsigned value field in a cursor position command indicates absolute cursor movement.

Relative motion specifies the distance to move *referenced from the current active position* (CAP) (see Figure 6-1). A signed (+/-) value field in a cursor position command indicates relative cursor movement.

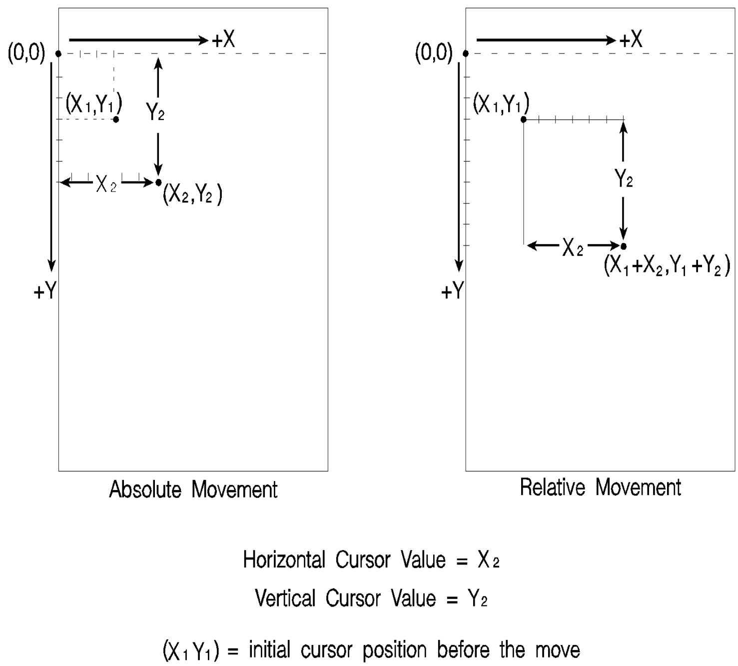


Figure 6-1 Absolute and Relative Cursor Positioning

Cursor Positioning Units

Cursor positioning is done in PCL coordinate system units. The units of the X-axis of the PCL coordinate system may be **PCL Units**, **decipoints**, or **columns**. The units of the Y-axis of the PCL coordinate system may be **PCL Units**, **decipoints**, or rows.

PCL Units

The current unit size used in PCL Unit moves is determined by the value specified in the Unit of Measure command, defining the number of units-per-inch used in the following commands:

- Vertical Cursor Position (PCL Units).
- Horizontal Cursor Position (PCL Units).
- Vertical Rectangle Size (PCL Units).
- Horizontal Rectangle Size (PCL Units).

In addition, the current unit of measure setting affects how cursor movement values are rounded, in turn affecting the result of the following commands:

- Horizontal Cursor Position (Columns).
- Horizontal Tab (HT control code).
- Space (SP control code).
- Backspace (BS control code).
- Bitmap Character Delta X (Delta X (SI), Chapter 11).

For more information, refer to the next section, "*Horizontal Cursor Positioning (Columns) Command*."

If no unit of measure value is specified, the default number of units-per-inch for PCL Unit moves (horizontal and vertical rectangle size, etc.) is one Unit equals 1/300 inch. This is true even when a different resolution (such as 600 dpi) is selected on the printer.

Decipoints

In PCL terminology, a decipoint is 1/720 inch or one-tenth of a PCL point (a PCL point is *exactly* 1/72 inch as opposed to a typographic point which is *approximately* 1/72 inch).

Columns & Rows

The width of a column is defined by the current horizontal motion index (HMI), as described under “Horizontal Motion Index (HMI) Command” in Chapter 5. The distance between rows is defined by the current vertical motion index (VMI), as described under “Vertical Motion Index (VMI) Command” in Chapter 5. HMI is the distance between consecutive characters. VMI is the distance between consecutive lines of text. HMI and VMI are described in more detail in Chapter 5.

HP-GL/2 has its own coordinate system and units. For additional information about the HP-GL/2 coordinate system and units, refer to Chapter 17, *An Introduction to HP-GL/2 Vector Graphics*.

Horizontal Cursor Positioning (Columns) Command

This Horizontal Cursor Positioning command moves the cursor to a new column on the current line.

^{E}C **& a # C**

= Number of Columns

Default = NA

Range = 0 - logical page right bound (valid to 4 decimal places)

The width of a column is defined by the current HMI.

Note The current unit of measure setting affects how HMI values are rounded. For example, if the unit of measure is set to 96 (one PCL Unit = 1/96 inch), then the HMI is rounded to the nearest 1/96 inch. If the unit of measure is set to 300 (one PCL Unit = 1/300 inch), the HMI is rounded to the nearest 1/300 inch.

A value field (#) with a plus sign (+) indicates the new position is to the right of and relative to the current cursor position; a minus sign (–) indicates the new position is to the left of and relative to the current cursor position. No sign indicates an absolute distance which is referenced from the left edge of the logical page. The first column within a line is column 0. This sequence ignores margins and can therefore be used to set the current active position (CAP) to any location along the current line.

If a request is made for a location outside the printer's logical page, the CAP is moved to the appropriate logical page limit.

Horizontal Cursor Positioning (Decipoints) Command

This Horizontal Cursor Positioning command moves the cursor to a new position along the horizontal axis.

$^{E}{}_{C}$ **& a # H**

= Number of Decipoints (1/720 inch)

Default = NA

Range = 0 - logical page right bound (rounded to the first decimal place)

A value field (#) with a plus sign (+) indicates the new position is to the right of and relative to the current cursor position; a minus sign (–) indicates the new position is to the left of and relative to the current cursor position. No sign indicates an absolute distance which is referenced from the left edge of the logical page. The left most position is 0 and the right most position is the right bound of the logical page.

If a request is made for a location outside the printer's logical page, the current active position (CAP) is moved to the appropriate logical page limit.

The value field is valid to two decimal places.

Horizontal Cursor Positioning (PCL Units) Command

This Horizontal Cursor Positioning command moves the cursor to a new position along the horizontal axis.

$_{E}{}_{C}$ *** p # X**

= Number of PCL Units

Default = NA

Range = 0 - logical page right bound

A value field (#) with a plus sign (+) indicates the new position is to the right of and relative to the current cursor position; a minus sign (–) indicates the new position is to the left of and relative to the current cursor position. No sign indicates an absolute distance which is referenced from the left edge of the logical page. The left most position is 0 and the right most position is the right bound of the logical page.

If a request is made for a location outside the printer's logical page, the current active position (CAP) is moved to the appropriate logical page limit.

Note The current unit size used in PCL Unit moves is determined by the value specified in the Unit of Measure command. If no other value is specified, the number of units-per-inch for PCL Unit moves is one unit equals 1/300 inch.

Horizontal Cursor Positioning Control Codes

Four control codes can be used to position the cursor horizontally on the current line. These control codes are explained below.

Note The distance which the cursor is moved by the Space (SP), Backspace (BS), and Horizontal Tab (HT) control codes is defined by the current HMI value. The current unit of measure setting affects how HMI values are rounded. For example, if the unit of measure is set to 96 (one PCL Unit = 1/96 inch), then the HMI is rounded to the nearest 1/96 inch. If the unit of measure is set to 300 (one PCL Unit = 1/300 inch), the HMI is rounded to the nearest 1/300 inch.

CR - Carriage Return

Moves the current active position (CAP) to the left margin on the current line. (Refer to “Line Termination Command” later in this chapter.)

SP - Space

Moves the current active position (CAP) to the right by one column position. Space may be a printable character or a control code. If a character is defined for the Space code, Space is printable; otherwise, it is a control code. For proportionally spaced fonts, a Space control code moves the cursor by the current HMI value; however, a printable space moves the cursor the width of the character. For fixed pitch fonts, a space, whether control code or printable, moves the cursor according to the HMI value.

BS - Backspace

Moves the current active position (CAP) left a distance equal to the width of the last printed symbol or space. If the active position is already at the left margin, no action is taken. If the cursor is currently beyond the right margin, BS positions the cursor just to the left of the right margin. When using fixed pitch fonts, the Backspace distance is defined by the current print pitch (HMI setting).

When using proportionally-spaced fonts, a single Backspace moves back to center the overstrike character. After printing the overstriking character, the cursor returns to its position prior to the Backspace. Multiple backspaces each move back the distance of the last printed symbol or space. For example, if “world” was printed with a proportional font and then 5 backspaces were performed, the distance moved back would be five times the width of the “d.”

HT - Horizontal Tab

Moves the current active position (CAP) to the next tab stop on the current line. The tab stops are at the left margin and every 8th column between the left margin and the right bound of the logical page. If the new horizontal position crosses the right margin, the new horizontal position is set to the right margin. If the current HMI value is 0, the command is ignored.

Vertical Cursor Positioning (Rows) Command

This Vertical Cursor Positioning command moves the cursor to a new line in the same column position.

$^{E}{}_{C}$ **& a # R**

= Number of Rows

Default = NA

Range = –32767 to 32767 (valid to 4 decimal places)

A value field (#) with a plus sign (+) indicates the new position is downward from and relative to the current cursor position; a minus sign (–) indicates the new position is upward from and relative to the current cursor position. No sign indicates the new position is absolute from the top margin. The top position, defined by the top margin, is 0 and the bottom position is determined by the bottom of the logical page.

Note Since the top margin can be changed using a printer command, the physical location of the point (0,0) may change. This affects the cursor position on the page.

If a request is made for a location outside the printer's logical page, the current active position (CAP) is moved to the appropriate logical page limit.

Vertical Cursor Positioning (Decipoints) Command

This Vertical Cursor Positioning command moves the cursor to a new position along the vertical axis.

^{E}C **& a # V**

\# = Number of Decipoints (1/720 inch)

Default = NA

Range = –32767 to 32767 (rounded to the first decimal place)

A value field (#) with a plus sign (+) indicates the new position is downward from and relative to the current cursor position; a minus sign (–) indicates the new position is upward from and relative to the current cursor position. No sign indicates an absolute distance from the top margin. The top position, defined by the top margin, is 0 and the bottom position is determined by the bottom of the logical page.

Note Since the top margin can be changed using a printer command, the physical location of the point (0,0) may change. This affects the cursor position on the page.

If a request is made for a location outside the printer's logical page, the current active position (CAP) is moved to the appropriate logical page limit.

Vertical Cursor Positioning (PCL Units) Command

This Vertical Cursor Positioning command moves the cursor to a new position along the vertical axis.

^{E}C *** p # Y**

= Number of PCL Units

Default = NA

Range = –32767 to 32767

A value field (#) with a plus sign (+) indicates the new position is downward from and relative to the current cursor position; a minus sign (–) indicates the new position is upward from and relative to the current cursor position. No sign indicates an absolute distance from the top margin. The top position, defined by the top margin, is 0 and the bottom position is determined by the bottom of the logical page.

Note Since the top margin can be changed using a printer command, the physical location of the point (0,0) may change. This affects the cursor position on the page.

If a request is made for a location outside the printer's logical page, the current active position (CAP) is moved to the appropriate logical page limit.

Note The current unit size used in PCL Unit moves is determined by the value specified in the Unit of Measure command. If no other value is specified, the number of units-per-inch for PCL unit moves is one unit equals 1/300 inch.

Half-Line Feed Command

The Half-Line Feed command moves the cursor to the same character position one half-line down. The distance moved for a Half-Line Feed is one-half of the current line spacing (defined by the last VMI or line spacing setting).

$^{E}{}_{C}$ **=**

Vertical Cursor Positioning Control Codes

Two control codes can be used to position the cursor vertically. These control codes are explained below.

LF - Line Feed

Advances the current active position (CAP) to the same horizontal position on the next line. The distance to the next line is defined by the current line spacing (defined by the last VMI or line spacing setting). (Refer to “Line Termination Command” later in this chapter.)

FF - Form Feed

Advances the current active position (CAP) to the same horizontal position at the top of the text area on the next page. (Refer to “Line Termination Command” later in this chapter.)

Line Termination Command

The Line Termination command controls the way the printer interprets CR, LF, and FF control characters. All CR, LF and FF control characters received after the Line Termination Command are interpreted as shown below.

E_C **& k # G**

\# = 0- CR=CR; LF=LF; FF=FF
1 - CR=CR-LF; LF=LF; FF=FF
2 - CR=CR; LF=CR-LF; FF=CR-FF
3 - CR=CR-LF; LF=CR-LF; FF=CR-FF

Default = 0

Range = 0-3

For example, if a value field of 1 is sent, the printer interprets each Carriage Return (CR) received as a **Carriage Return (CR) and Line Feed (LF)** control code. A Line Feed or Form Feed would be sent as is.

If a value of 3 is sent, the printer interprets each Carriage Return (CR) received as a **Carriage Return (CR) and Line Feed (LF)**; it interprets each Line Feed (LF) received as a **Carriage Return (CR) and Line Feed (LF)**; and it interprets each Form Feed (FF) received as a **Carriage Return (CR) and Form Feed (FF)**.

Push/Pop Cursor Position Command

The Push/Pop Cursor Position command allows the current cursor position to be stored and recalled.

^{E}C **& f # S**

\# = 0 - Push (Store cursor position)
1 - Pop (Recall a cursor position)

Default = 0

Range = 0, 1 (Values outside range are ignored)

A value field of 0 pushes the cursor position onto the stack, leaving the current position unaffected. A value field of 1 pops the position from the stack, restoring it as the current cursor position.

Note The last item pushed is the first item popped.

Twenty positions may be pushed. If you try to save more than 20 positions, the command is ignored. If you try to restore more positions than were pushed, the command is ignored. A printer reset restores the current active position stack to the top (all saved positions are discarded).

The positions stored in the stack are not changed with an orientation change. Therefore, the positions are relative to the top left corner of the current orientation. Also, a position pushed in one orientation and popped in another can result in a position that is outside the logical page. If the position popped is outside the current logical page, the position is moved to the appropriate logical page limit.

7 Fonts

Introduction

A font is a group of symbols that have similar characteristics. A font is described by its **symbol set**, **spacing**, **height**, **pitch**, **style**, **stroke weight**, **typeface** and **orientation**.

A typical document is printed using several fonts. A large font may be used for the title and chapter headings of a document, a standard size font may be used for the body of the document, and key words or phrases may be highlighted, using a bold or italic font.

For example, this text is printed using a Century Schoolbook typeface; its **height** is 10 point, its style is upright, and its **stroke weight** is medium. Examples of different fonts are shown in Figure 7-1.

Courier 12 pitch 10 point
abcdefABCDEF12345&%!?

CG Times 12 point
abcdefABCDEF12345&%!?

CG Times Bold Italic 14 point
abcdefABCDEF12345&%!?

Univers Medium 14 point
abcdefABCDEF12345&%!?

Univers Bold 24 point
abcdefABCDEF12345&%!?

Figure 7-1 Font Samples

A font must be selected for printing by the user. One font is selected at a time. It is selected by identifying the specific characteristics of the font. Font selection commands identify font characteristics to the printer (refer to *Summary of Font Selection by Characteristic* for detailed font selection information).

PCL 5 printers feature scalable fonts. With the addition of this feature, the printer has two font formats available: bitmap and scalable. A bitmap font is available in its one, defined size only. A scalable font, on the other hand, can be selected (scaled) for a range of sizes (refer to “Bitmap Fonts and Scalable Typefaces” later in this chapter for additional information).

Font Sources

A number of fonts (and typefaces, as described later) are supplied with the printer. These fonts reside in permanent ROM (read only memory), and are referred to as **internal fonts**. Additional fonts can be added easily by inserting font cartridges or SIMM modules into the printer, or downloading them from the host computer.

A **cartridge font** plugs into a font cartridge slot on the printer. **SIMM font** modules plug into a printed circuit board inside the printer. These ROM-based fonts are always available (as long as the cartridge or SIMM module is installed). A variety of font products may be purchased from Hewlett-Packard or other vendors. Refer to your Hewlett-Packard Accessories and Supplies Brochure for a list of HP’s font products.

Soft fonts are supplied as files on flexible disk transferred (downloaded) into the printer’s user (RAM) memory. Once a soft font has been downloaded into the printer’s RAM, it may be selected for printing.

Symbol Set

Symbol set identifies the specific collection of symbols provided by a font. Each symbol set is defined with a specific application in mind. For example, the legal and math symbol sets were designed to support legal and scientific applications. The following figure shows two common symbol sets, PC-8 and Roman-8. The PC-8 symbol set contains some special symbols and line draw characters not included in the Roman-8 symbol set, while the Roman-8 set contains European characters not contained in the PC-8 symbol set.

Note User-defined symbol sets are supported by some HP LaserJet printers. See Chapter 10 for more information.

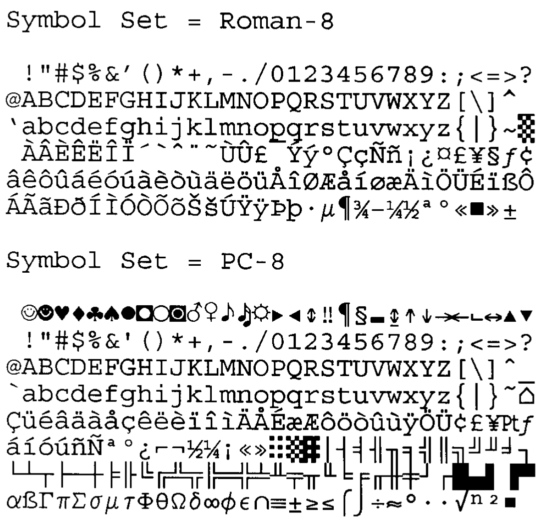


Figure 7-2 Symbol Sets

Spacing

Another characteristic that differentiates fonts is spacing. Fonts have either fixed or proportional spacing. Fixed-spaced fonts (Figure 7-3) are those in which the inter-character spacing is constant. Proportionally-spaced fonts (Figure 7-4) are those in which the inter-character spacing varies with the natural shape of a character.

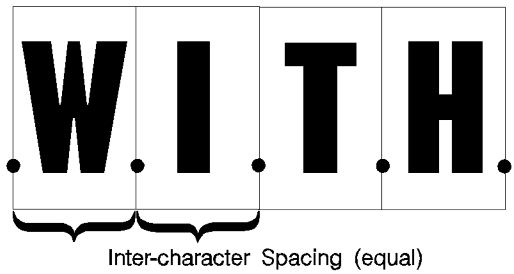


Figure 7-3 Fixed Spacing

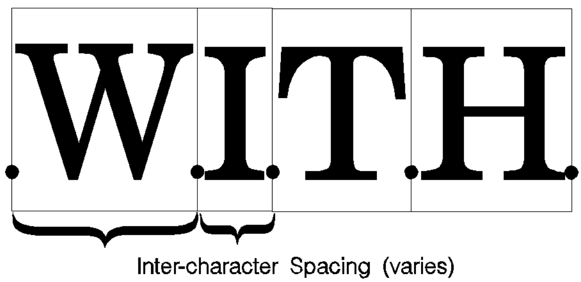


Figure 7-4 Proportional Spacing

Pitch

Pitch describes the number of characters printed in a horizontal inch. Pitch only applies to fixed-spaced fonts, since the number of characters per inch varies for proportional fonts.

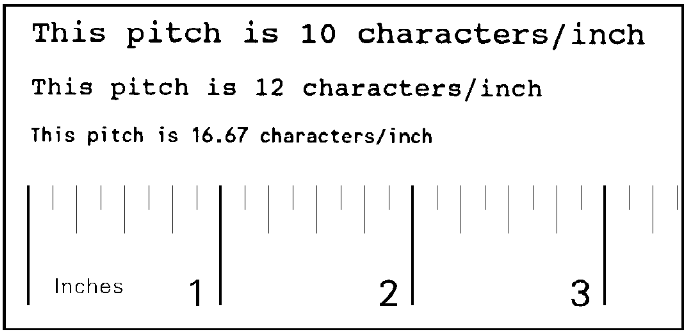


Figure 7-5 Pitch

Height

The height of a font is the measurement of the body of the type in PCL points. A PCL point is 1/72 inch in bitmap fonts, and approximately 1/72 inch in scalable fonts. The body of the type is slightly larger than the distance from the bottom of a descender to the top of an unaccented capital letter.

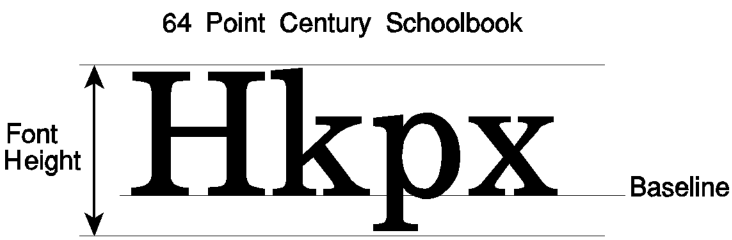


Figure 7-6 Height

This loose measure from near the bottom of a descender to just above the top of an unaccented capital letter is sometimes referred to as the “Em.”

Style

Style is defined by three characteristics: posture (upright, italic), width (condensed, normal, expanded, etc.), and structure (solid, outline, shadow etc.). Examples of upright and italic styles are shown.



Figure 7-7 Style

Stroke Weight

Stroke weight describes the thickness of the strokes that compose characters. Examples of medium and bold stroke weights are shown in the figure below.



Figure 7-8 Stroke Weight

Typeface Family

Typeface identifies the design of the symbols of the font. Each typeface family has unique and distinguishing design characteristics. The following example shows typefaces from various typeface families.

Brush

Uncial

Dom Casual

CG Palacio

University Roman

Futura Book II

Garamond Kursiv

ITC Souvenir Light

ITC Benguiat Book

Microstyle Extended

Figure 7-9 Typeface

Orientation

Orientation defines the position of the logical page with respect to the physical page as shown in Figure 7-10.

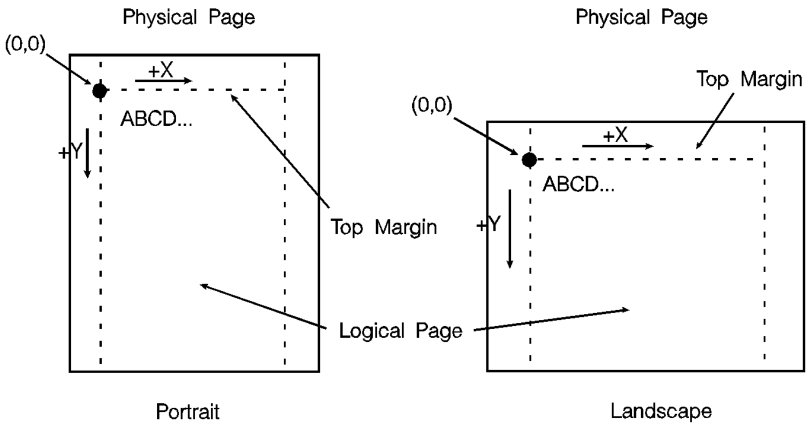


Figure 7-10 Orientation

The HP LaserJet IID, IIP, 2000, and all PCL 5 LaserJet printers automatically rotate fonts to the current orientation (all fonts are available in all four orientations). (Earlier printers required fonts in the orientation which matched the orientation of the page. Thus, orientation is not as important as it once was.)

The orientation of a font is still a consideration when the amount of user memory (RAM) is a concern. Internal and other ROM-based fonts consume very little user memory. On some printers, downloaded fonts, scaled fonts, and rotated fonts are stored entirely in RAM. For bitmap fonts, selecting a font with the current logical page orientation saves RAM space on some printers.

Bitmap Fonts and Scalable Typefaces

There are two basic formats of fonts used by HP PCL 5 printers: bitmap (Figure 7-11) and scalable (Figure 7-12). Earlier HP LaserJet printers supported only bitmap fonts. Bitmap fonts have a fixed bit-pattern for each character. The size of the character is fixed, depending on the bit-pattern. Scalable typefaces, on the other hand, provide an “outline” for the characters. This “outline” can be scaled by the PCL 5 printers to produce a large range of character sizes.

There is a difference, between a scalable typeface and a scalable font. A bound, **scalable font** is a group of “outline” characters limited to one specific symbol set. For a scalable font, the symbol set, spacing, style, stroke weight, and typeface characteristics are all fixed, and size is variable (since it is scalable). A s**calable typeface**, on the other hand, is a grouping of “outline” characters of a specific typeface which can produce multiple symbol sets. For a scalable typeface, spacing, style, stroke weight, and typeface characteristics of the font are all fixed, symbol set and size are variable.

Note Scalable fonts and scalable typefaces are selected for printing in the same manner as bitmap fonts; no additional selection is required (refer to Summary of Font Selection by Characteristic for font selection information).

Some scalable typefaces are provided with the printer (for example: CG Times and Univers). Additional scalable typefaces can be obtained on disk, cartridge or SIMM modules.

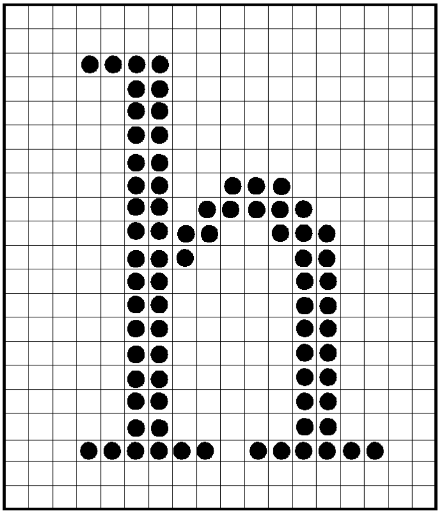


Figure 7-11 Bitmap Character

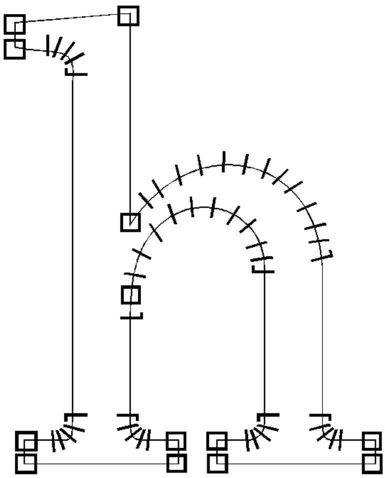


Figure 7-12 Scalable Character

Internal Fonts

Internal fonts are those fonts that are provided with the printer. Both internal bitmap and scalable font formats are provided. Internal bitmap and scalable fonts and symbol sets for current models of HP LaserJet family printers are listed in Chapter 2 of the *PCL 5 Comparison Guide*. Font and symbol set listings for earlier printer models can be found in the *User's Manual* for each printer.

Special Effects

HP PCL 5 printers allow you to create special effects when printing characters. These effects are achieved through the use of the print model feature, or through the use of HP-GL/2 vector graphics (refer to Chapter 13, *The PCL Print Model* or to the HP-GL/2 information in Chapters 17-23).

The print model provides a simple means for printing patterned or shaded characters using the printer's predefined cross-hatch/shading patterns, or user-defined patterns. HP-GL/2 vector graphics provide the additional ability to print characters in any direction (angle) on a page, and to print outlined characters. HP-GL/2 also allows anisotropic (non-linear) scaling of scalable fonts which produces characters that are stretched in one direction.

8 PCL Font Selection

Introduction

Several characteristics identify a font (as described in **Chapter 7**, *Fonts*). Font characteristic selection commands, described in this chapter, are used to specify the desired font characteristics for printing. Commands are included for the following characteristics: symbol set, spacing, pitch, height, style, stroke weight, and typeface family.

The printer maintains a **font select table** in its operating code that contains the characteristic values of the current font. Whenever the printer receives a font select command (escape sequence) specifying a new characteristic value, the printer records that characteristic in the table. After the table is updated (receives new characteristic values), and text is ready to be printed, the printer performs a **font select**. The printer searches the available fonts and scalable typefaces to select one that matches (or most closely matches) the characteristics as listed in the font select table.

Note **A font must be in the printer to be selected for printing!**

Font Selection Priority

The printer selects a font based on a prioritization of its design characteristics, then its resolution, then its physical location in the printer, and finally, its orientation. Font selection priority considerations are shown in the following list:

Font Priority Considerations	
Symbol Set	highest
Spacing	|
Pitch	|
Height	|
Style	|
Stroke Weight	|
Typeface Family	|
Resolution[1]	|
Location[2]	|
Orientation	lowest

1. Bitmap fonts designed at 600 dpi are not available for selection at 300 dpi. In 600 dpi mode, font priority is as follows: 600 dpi bitmap, scalable, 300 dpi bitmap.
2. Although location is not a font characteristic, it is a font selection consideration.

When selecting a font, the printer compares the highest priority characteristic in the font select table to the corresponding characteristic of the available fonts. If only one font is available that matches, that font is selected. If several fonts match, the printer compares the next highest priority characteristic to the corresponding characteristic of the available fonts and so on down the list. When only one font remains, that font is selected. However, if after comparison of all the font design characteristics, more than one font still remains, then the resolution and location are considered.

There are four locations where a font may be stored: printer ROM (Read Only Memory), SIMM module ROM, cartridge ROM, and printer RAM (random access memory; user memory). These font locations are shown below, listed from the highest to lowest priority. The font that matches the characteristics is selected from the highest priority location.

Priority of Locations	
Soft Font (Lowest ID first)	Highest
Cartridge Font[1]	|
SIMM Font	|
Internal Font	Lowest

1. In printers with two cartridge slots, one slot has priority over the other. Refer to Appendix E of the *PCL 5 Comparison Guide* for cartridge slot priority information for the different HP LaserJet printers.

Note

In 600 dpi mode: A 600 dpi font has priority over a 300 dpi font. For example, a 600 dpi bitmap soft font is highest, then a scalable soft font, followed by a 300 dpi bitmapped soft font.

Finally, for bitmap fonts, the orientation of a font is considered. If there are two fonts which are similar in all the above characteristics and which reside at the same location, the font with the orientation that matches the orientation of the page is selected. If only one font remains and its orientation is different than the current page, the printer rotates the font to the orientation of the page. (“Summary of Font Selection by Characteristic” on page 23, later in this chapter, summarizes font selection by characteristic.)

Font Select Table

The initial font specification in a job should be made using all of the font characteristics.

To select a Roman-8, fixed-spaced, 10 pitch, 12 point, upright, bold, Courier font, for the current page orientation, specify each of the characteristics using font selection escape sequences. Once the characteristics have been specified, the font select table appears as follows:

Symbol Set	Roman-8
Spacing	Fixed
Pitch	10 cpi
Height	12 point
Style	Upright
Stroke Weight	Bold
Typeface Family	Courier

To subsequently select a font with the same characteristics differing only in one aspect, only the single characteristic must be specified. For example, to select a font differing only in stroke weight (in this case, medium rather than bold), the printer's font select table could be changed as follows:

Symbol Set	Roman-8
Spacing	Fixed
Pitch	10 cpi
Height	12 point
Style	Upright
Stroke Weight	**Medium**🡐
Typeface Family	Courier

At a minimum, only the characteristics of the new font that differ from those of the previously designated font must be sent (the short font selection method). However, **HP recommends that all of the characteristics be sent to ensure that the correct font is selected**.

Note PCL 5 printers can print any number of distinct fonts per page, limited only by available memory.

Primary and Secondary Fonts

The printer maintains two independent font select tables for use in selecting a primary font and a secondary font. All of the characteristics previously described apply to both tables. This provides access to two distinct fonts, only one of which is selected at a given time. To alternate between the primary and the secondary font, the control codes “SI” (Shift In; ASCII 15) is used to designate **primary** and “SO” (Shift Out; ASCII 14) is used to designate **secondary.**

The factory default state is **primary** font designated.

Font Resolution

With the introduction of the LaserJet 4 printer, fonts can be printed at 600 dpi resolution. All scalable fonts automatically print at either 300 or 600 dpi resolution. A bitmapped font which was designed at 300 dots-per-inch can be printed on the LaserJet 4 printer at 600 dpi. However, a bitmapped font which was designed at 600 dpi is not available for selection at 300 dpi resolution.

Symbol Set Command

The Symbol Set command identifies the specific set of symbols in a font. “Symbols” are the alphanumeric, punctuation, or any other printable characters or symbols which may be included.

$^{E}_{C}$ **(ID** Primary Symbol Set Command

ID = Symbol Set ID value (see Appendix C in the *PCL 5 Comparison Guide*)

$^{E}_{C}$ **) ID** Secondary Symbol Set Command

ID = Symbol Set ID value (see Appendix C in the *PCL 5 Comparison Guide*).

Default = 8U

Range = N/A

If the specified symbol set does not exist, Roman-8 is selected (However, the specified symbol set is written into the font select table.)

Notes

The factory default primary and secondary symbol set is Roman-8. However, you may select a user default symbol set from the printer control panel (see the printer *User’s Manual*).

If the font is a scalable typeface, symbol set is determined from the values contained in the printer’s font selection table. To specify a different symbol set, send a symbol set selection command **prior** to the Font Selection ID command. (Also see “Font Selection by ID Command,” later in this chapter.)

A few symbol sets are listed below. For a more complete list, refer to Appendix C in the *PCL 5 Comparison Guide*.

Typical Symbol Set Values	
Symbol Set Name	**Symbol Set ID**
ISO 69: French	1F
ISO 8859-1 Latin 1 (ECMA-94)	0N
ISO 6:ASCII	0U
Legal	1U
Roman-8	8U
PC-8	10U
3 of 9 Barcode	0Y
Windows 3.1 Latin 1 (ANSI)	19U

Note

User-defined symbol sets are supported in some HP LaserJet printers. To specify a user-defined symbol set, use the symbol set ID value as defined by the **Symbol Set ID Code Command**. See Chapter 10 for more information.

Example

To specify ASCII as the symbol set for the primary font, send:

$^{E}{}_{C}$(ØU

To specify Roman-8 as the symbol set for the secondary font, send:

$^{E}{}_{C}$)8U

7-bit ISO Symbol Sets

The HP LaserJet printers provide several 7-bit ISO (International Organization for Standardization) or “keyboard” symbol sets to support European languages. Each ISO symbol set is a unique ordering of symbols contained within the Roman-8 symbol set (see Appendix B in the *PCL 5 Comparison Guide*). The printer automatically generates the requested ISO font from an HP Roman-8 font.

Spacing Command

Inter-character spacing can be specified as either proportional or fixed.

$^{E}{}_{C}$ (s # P - Primary spacing

$^{E}{}_{C}$) s # P - Secondary spacing

\# = 0 - Fixed spacing
1 - Proportional spacing

Default = 0

Range = 0, 1 (values outside the range are ignored)

When proportional spacing is specified and a proportionally-spaced font is not available (in the requested symbol set), a fixed pitch font with the current pitch specification is selected. If fixed spacing is specified but is not available, a proportional-spaced font is selected and the pitch characteristic is ignored.

For fixed-spaced bitmap fonts, both pitch and height (point size) are used for selection of font character size. However, for fixed-spaced scalable fonts, only pitch is used. For proportional bitmap and scalable fonts, only height is used for selection of font character size.

The user default primary and secondary spacings are implicitly set by selection of a user default font from the printer's control panel (refer to the printer *User's Manual*).

Example

To specify proportional spacing for the primary font, send:

$^{E}{}_{C}$(s1P

To specify fixed spacing for the secondary font, send:

$^{E}{}_{C}$)s0P

Pitch Command

The Pitch command designates the horizontal spacing of a fixed- spaced (bitmap or scalable) font in terms of the number of characters per inch. This characteristic is ignored when selecting a proportionally-spaced (bitmap or scalable) font, but is saved in the font select table and available when a fixed-spaced font is selected.

$^{E}{}_{C}$ **(s # H** - Primary pitch

$^{E}{}_{C}$ **) s # H** - Secondary pitch

= # = Pitch in characters/inch

Default = 10

Range = 0.00

The value field (#) is valid to two decimal places.

If a pitch is specified that is not available, the next greater available pitch is selected. If no greater value is available, the closest available lesser value is selected.

The factory default primary and secondary pitches are ten characters per inch.

The user default primary and secondary pitches are implicitly set by selection of a users default font from the printer's control panel (refer to the printer *User's Manual*).

The range of valid pitch selections for a fixed-spaced scalable font is 576 to .10 characters/inch, however, not all valid pitches are available, since the pitch value is actually converted to a corresponding point size (height) value which is scaled by the printer. The effective pitch ranges are thus limited by height constraints.

The lower end of the pitch range is limited as a result of the font height limitation of 999.75 points. For example, the smallest available pitch for the internal Courier typeface would be about 0.12.

The upper end of the pitch range is similarly limited by the minimum recommended font height of 4 points. For Courier, this translates to a maximum recommended pitch of 30 (30 cpi), while for Letter Gothic the maximum recommended pitch is 36. When requested pitch values are outside of HP's recommended limits, unsatisfactory results can occur.

The following formula can be used to as a rule of thumb for computing a maximum recommended pitch:

$$\frac{1}{\text{Contour Width (percent of Em)} \times 4 \text{ (points)} \div 72 \text{ (points/inch)}}$$

The character ("contour") width in the above formula is expressed as a percentage of an Em. For example, the width of characters in Courier is very close to 60% of an Em, and $30 = 1 \div (0.64 \div 72)$. (The width of characters in Letter Gothic is approximately 50% of an Em. For other fonts, refer to the font metric data supplied by the font vendor.)

Note

If a scalable fixed-space font is selected using an ID number, send the Pitch command to specify the size; otherwise, the size is determined by the pitch characteristic value of the former font (as listed in the font select table) See "Font Selection by ID Command" later in this chapter for more information.

Example

To specify 10 pitch for the primary font, send:

$^{E}{}_{C}$(s10H

To specify 16.66 pitch for the secondary font, send:

$^{E}{}_{C}$)s16.66H

Height Command

The Height command specifies the height of the font in points. This characteristic is ignored when selecting a fixed-spaced scalable font; however, the value is saved and available when a bitmap font or a proportionally-spaced scalable font is selected.

$^{E}_{C}$ **(s # V** - Primary Height

$^{E}_{C}$) **s # V** - Secondary Height

= Height in points

Default = 12

Range = 0.25 - 999.75

The value field (#) is valid to two decimal places. If the requested height is unavailable, the closest height is selected. All bitmap fonts whose heights are within a quarter point of the specified height are considered to have the specified height. For scalable fonts the value field is from .25 to 999.75 points in increments of 0.25 point (values are rounded to the nearest quarter point).

The factory default primary and secondary heights are 12 point. In PCL bitmap fonts, a point is 1/172 (0.01389) inch. For scalable fonts, the definition of a point varies in TrueType a point is 1/172 inch, while Intellifont fonts have 72.307 points to the inch.

The user default primary and secondary heights are implicitly set by selection of a user default font from the printer's control panel (refer to the printer *User's Manual*).

Note If a proportional-spaced scalable font is selected using an ID number, send the Height command to specify the point size otherwise, the size is determined by the height characteristic value of the former font (as listed in the font select table) See “Font Selection by ID Command” later in this chapter for more information.

Example

To specify a height of 12 points for the primary font, send:

$^{E}{}_{C}$(s12V

To specify a height of 14.4 points for the secondary font, send:

$^{E}{}_{C}$)s14.4V

If the above sequence was used for selection of a scalable font, the actual font would be scaled to 14.5 points.

Style Command

The Style command identifies the posture of a character, its width, and structure of the font symbols.

$^{E}{}_{C}$ **(s # S** - Primary Style

$^{E}{}_{C}$ **) s # S** - Secondary Style

Default = 0

Range = 0 - 32767 (values greater than 32767 are set to 32767)

Style values for the most common typefaces are listed in Table 8-1. Additional style values may also be obtained from the related font documentation provided with HP's font products.

Table 8-1 Common Font Styles

Value	Font Styles
0	(upright, solid)
1	italic
4	condensed
5	condensed italic
8	compressed, or extra condensed
24	expanded
32	outline
64	inline
128	shadowed
160	outline shadowed

Notes

With the introduction of the HP LaserJet IID printer, Hewlett-Packard expanded the style values (in the Font Header style value field) from a one-byte to a two-byte value field, expanding the style range from 0-255 to 0-32767. This expansion allows for additional styles.

Style values can be obtained by calculating the “Style Word” as described under “Style MSB” in Chapter 11.

For selecting style, an exact match is required. If there is no match, this characteristic is ignored, but stored in the font select table, available for the next selection.

Example

To specify an upright style for the primary font, send:

$^{E}{}_{C}$(s0S

To specify an italic style for the secondary font, send:

$^{E}{}_{C}$)s1S

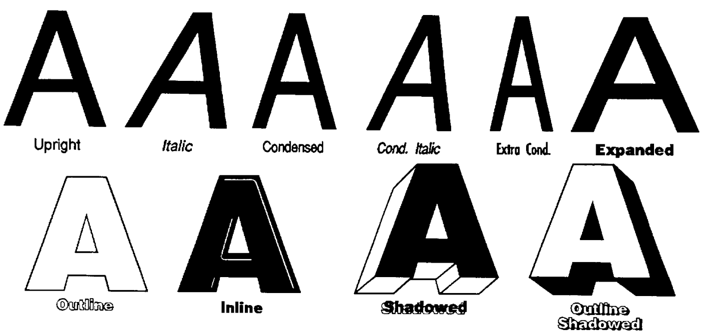


Figure 8-1 Common Font Styles

Stroke Weight Command

The Stroke Weight command designates the thickness of the strokes that compose the characters of a font.

$^{E}{}_{C}$ **(s # B** - Primary stroke weight

$^{E}{}_{C}$ **) s # B** - Secondary stroke weight

Default = 0

Range = - 7 to 7 (less than -7 maps to -7; greater than 7 maps to 7)

The value field (#) specifies the thickness of the strokes used in the design of the font. The supported stroke weight values are –7 through 7. The thinnest font available is –7; the thickest font available is +7. The standard stroke weight for a medium font is 0; the standard stroke weight for a bold font is 3; the standard stroke weight for a light font is –3.

Table 8-2 Stroke Weights

Value (#)	Typeface
-7	Ultra Thin
-6	Extra Thin
-5	Thin
-4	Extra Light
-3	**Light**
-2	Demi Light
-1	Semi Light
0	**Medium, Book, or Text**
1	Semi Bold
2	Demi Bold
3	**Bold**
4	Extra Bold
5	Black
6	Extra Black
7	Ultra Black

If the specified stroke weight is greater than or equal to 0 and is not available, the next thicker available stroke weight is selected. If no thicker stroke weight is available, the closest available thinner stroke weight is selected.

If the specified stroke weight is less than zero and is not available, the next thinner available stroke weight is selected. If no thinner stroke weight is available, the closest available thicker stroke weight is selected.

The factory default primary and secondary stroke weights are zero (medium).

The user default primary and secondary stroke weights are implicitly set by selection of a user default font from the printer's control panel (refer to the printer *User's Manual*).

Example

To specify a bold stroke weight for the primary font, send:

$^{E}{}_{C}$(s3B

To specify a medium stroke weight for the secondary font, send:

$^{E}{}_{C}$)s0B

Note Many typefaces were designed for advertising use, and a "medium" was used to describe the standard treatment. Later, additional treatments were designed for text use. Therefore, the typeface treatment designation "medium" may not always take a PCL value of 0. This weight value may be assigned to "book" or "text" treatment instead.

Typeface Family Command

The Typeface Family command designates the design of the font.

$^{E}{}_{C}$ **(s # T** - Primary typeface family

$^{E}{}_{C}$ **) s # T** - Secondary typeface family

\# = Typeface family value (see Appendix C in the *PCL 5 Comparison Guide* for typeface values).

Default = [1]4099, Body Text

Range = [1]0 - 65535 (values greater than 65535 are set to 65535)

If the value field (#) specifies a typeface that is unavailable, this characteristic is ignored during font selection.

The factory default primary and secondary typefaces are Courier.

The user default primary and secondary typefaces are implicitly set by selection of a user default font from the printer's control panel (refer to the printer *User's Manual*).

There is some variation in how font selection occurs between HP LaserJet models. The typeface selection compatibility for two types of values is identified for various HP LaserJet printers in Chapter 2 of the *PCL 5 Comparison Guide*.

Note Use the typeface family values, listed in Appendix C in the *PCL 5 Comparison Guide*, for future typeface selection.

1. These values are not applicable to all HP LaserJet family printers. See the *PCL 5 Comparison Guide* for specifics.

Some typeface (two-byte) family values are listed below. For a complete listing of typeface family and base values, refer to Appendix C in the *PCL 5 Comparison Guide*.

Sample Typeface Values	
FamilyValue	**Typeface Family**
0	Line Printer
16602	Arial
4168	Antique Olive
4127	ITC Avant Garde
4119	CG Century Schoolbook
4101	CG Times
4148	Univers

Example

To specify CG Times as the typeface family for the primary font, send:

$^{E}{}_{C}$**)s4101T**

To specify Line Printer as the typeface family for the secondary font, send:

$^{E}{}_{C}$**)s0T**

Orientation

The Orientation command ($^{E}{}_{C}$**&l#O**) designates the position of the logical page with respect to the physical page. Earlier printers could only print bitmap fonts and raster graphics in the orientation for which they were designed. However, the HP LaserJet IID, IIP, 2000, and all PCL 5 HP LaserJet printers have the capability to automatically rotate bitmap fonts and raster graphics to match the page orientation; therefore, all fonts are available in all four page orientations and print directions. Whenever a scalable font is selected, it is created in the current orientation for printing. Refer to “Logical Page Orientation Command” and “Print Direction Command” in Chapter 5 for more information.

Font Selection Examples

Bitmap, Fixed-Spaced Font

This example illustrates how to select a primary, bitmap, Line Printer, fixed-spaced font with the following characteristics (note that all of the font characteristics are specified):

Table 8-3

CHARACTERISTIC	VALUE	ESCAPE SEQUENCE
Symbol set	ASCII	**$^{E}_{C}$(0U**
Spacing	Fixed	**$^{E}_{C}$(s0P**
Pitch	16.66 cpi	**$^{E}_{C}$(s16.66H**
Height	8.5 point	**$^{E}_{C}$(s8.5V**
Style	Upright	**$^{E}_{C}$(s0S**
Stroke weight	Medium	**$^{E}_{C}$(s0B**
Typeface family	Line Printer	**$^{E}_{C}$(s0T**

The following escape sequences can be sent to the printer to select a primary font with the above characteristics:

$^{E}_{C}$(0U$^{E}_{C}$(s0P$^{E}_{C}$(s16.66H$^{E}_{C}$(s8.5V$^{E}_{C}$(s0S$^{E}_{C}$(s0B$^{E}_{C}$(s0T

The previous sequence can be shortened by combining sequences that have the same two characters following the $^{E}_{C}$ character:

$^{E}_{C}$(0U$^{E}_{C}$(s0p16.66h8.5v0s0b0T

Scalable, Proportional-Spaced Font

This example illustrates how to select a primary, scalable, CG Times, proportional-spaced font with the following characteristics (note that all of the font characteristics are specified except pitch which is not required for a proportional font):

CHARACTERISTIC	VALUE	ESCAPE SEQUENCE
Symbol set	ASCII	$^{E}_{C}$**(0U**
Spacing	Proportional	$^{E}_{C}$**(s1P**
Height	14.25 point	$^{E}_{C}$**(s14.25V**
Style	Upright	$^{E}_{C}$**(s0S**
Stroke weight	Bold	$^{E}_{C}$**(s3B**
Typeface family	CG Times	$^{E}_{C}$**(s4101T**

The following escape sequences can be sent to the printer to select a primary font with the above characteristics:

$^{E}_{C}$**(0U**$^{E}_{C}$**(s1P**$^{E}_{C}$**(s14.25V**$^{E}_{C}$**(s0S**$^{E}_{C}$**(s3B**$^{E}_{C}$**(s4101T**

Combining the above sequences results in:

$^{E}_{C}$**(0U**$^{E}_{C}$**(s1p14.25v0s3b4101T**

Notes

If an escape sequence does not contain a value field, the printer assumes a value of zero; therefore, the command $^{E}_{C}$**(sB** can be sent to the printer instead of $^{E}_{C}$**(s0B**.

Sending shortened font selection commands can result in selection of an unexpected font. This is due to failure to track previously specified characteristics and their selection priority in relation to the current font selection. Thus, **it is recommended that all of the characteristics be sent** to ensure that the correct font is selected.

Summary of Font Selection by Characteristic

The following summarizes the procedure the printer uses to select a font. Selection by characteristic is an elimination process. The nine steps are performed in the following order:

Note

When the printer is in 300 dpi mode, any 600 dpi bitmaps are eliminated before the selection process begins.

1 **Symbol Set** - if the specified symbol set exists, that symbol set is selected; otherwise, Roman-8 is selected.

2 **Spacing** - if proportional spacing is specified and available, proportional spacing is selected. If proportional spacing is specified but is not available, fixed spacing is selected in the current pitch. (A proportionally-spaced font is always available in PCL 5 printers, but it may not be available in the specified symbol set.)

3 **Pitch** - applies only to fixed spaced fonts. If fixed spacing is specified and available, fixed spacing in the specified pitch is selected.

   Bitmap Fonts: For a fixed-space bitmap font, if the specified pitch is not available, the next greater available pitch is selected. If no greater pitch is available, the closest available lesser pitch is selected. If fixed spacing is specified but is not available, a proportional-spaced font is selected and the pitch characteristic is ignored.

   Scalable Fonts: For a fixed-spaced scalable font, the pitch is used to calculate the appropriate height. The Height selection command is not required. The printer calculates the appropriate height to correspond to the pitch. The user's height request is recorded in the printer's font select table for later font selections, but is ignored for this selection.

4 **Height** - the closest height available from the remaining fonts is selected. The closest height is in terms of absolute difference. All bitmap fonts whose heights are within a quarter point of the specified height are considered to have the specified height.

Note

For proportionally-spaced scalable fonts, any specified height is available to the nearest quarter point. For fixed-spaced scalable fonts, the designated height is recorded, and the height is calculated from the requested pitch.

5 **Style** - if the specified style is available in the remaining fonts, that style is selected; otherwise, this characteristic is ignored.

6 **Stroke Weight** - if the specified stroke weight is available in the remaining fonts, that stroke weight is selected.

If the specified stroke weight is greater than or equal to 0 and is not available, the next thicker available stroke weight is selected. If no thicker stroke weight is available, the closest available thinner stroke weight is selected.

If the specified stroke weight is less than 0 and is not available, the next thinner available stroke weight is selected. If no thinner stroke weight is available, the closest available thicker stroke weight is selected.

7 **Typeface Family** - if the requested typeface is available in the remaining fonts, that typeface is selected; otherwise, this characteristic is ignored.

8 **Location** - if after performing all the preceding steps, more than one font remains, the available font from the highest priority font location is selected. The priority of the font locations are:

Priority of Locations	
Soft Font (Lowest ID first)	Highest
Cartridge Font[1]	|
SIMM Font	|
Internal Font	Lowest

1. In printers with two cartridge slots, one slot has priority over the other. Refer to Appendix E of the *PCL 5 Comparison Guide* for cartridge slot priority information for the different HP LaserJet printers.

9 **Orientation** - for bitmap fonts the last criteria considered for the selection is its orientation. If two fonts still remain and match in all the above characteristics except orientation, that font which matches the current page orientation is selected.

If there is a soft font (highest priority location) available that matches all selection characteristics, but is not in the current orientation, and there is an identical font available in a cartridge or internal font (lower priority location) that is in the current orientation, the soft font is selected and rotated.

Font Selectionby ID Command

Soft fonts can be specified using their associated ID numbers. (ID numbers are assigned to soft fonts using the Font ID command described in Chapter 9, *Font Management*).

$^{E}{}_{C}$ **(# X** - Designates soft font # as primary

$^{E}{}_{C}$ **) # X** - Designates soft font # as secondary

\# = font ID number

Default = 0

Range = 0 - 32767

If the designated font is present, the font is selected as the primary/secondary font and all primary/secondary font characteristics in the printer's Font Select Table are set to those of the selected font. However, if the selected font is proportionally spaced, the pitch characteristic is not changed.

If the designated font is not present, the current font is retained.

Notes

If a scalable font is selected using an ID number, send the Height or Pitch command (Height for proportional, Pitch for fixed) to specify the height or pitch; otherwise, the characteristic is determined by the value of the former font (as listed in the font select table).

If the font is a scalable typeface, symbol set is determined from the values contained in the printer's font selection table. To specify a different symbol set, send a symbol set selection command **prior** to the Font Selection ID command.

For shared or multi-user environments, Hewlett-Packard recommends that soft fonts be selected by characteristics rather than ID number.

Examples

To specify the font associated with ID number 7 as the primary font, send:

$^{E}{}_{C}$(7X

To specify the font associated with ID number 5 as the secondary font, send:

$^{E}{}_{C}$)5X

Select Default Font Command

The Default Font command sets all of the font characteristics to those of the user (control panel selected) default font.

$^{E}{}_{C}$ (3 @ - Default primary font characteristics

$^{E}{}_{C}$) 3 @ - Default secondary font characteristics

Note If the user default font is a proportionally-spaced font, the pitch characteristic is not affected by the default font command.

HP-GL/2 Font Selection

In addition to selecting fonts using the PCL font selection commands, fonts can also be selected and printed in HP-GL/2 mode using the HP-GL/2 label commands (refer to Chapter 23, *Character Group*). The HP-GL/2 font selection commands allow you to label vector graphic images and to create some special effects with fonts not otherwise available. These special effects include printing outline fonts from fonts which are not outline style, printing mirror-images of fonts, and printing fonts on any angle on the logical page. Fonts can also be scaled using HP-GL/2 vector graphics, however this font scaling method is rather program intensive and not the recommended method (refer to "HP-GL/2 Vector Graphics" in Chapter 25 for additional information).

Transparent Print Data Command

The Transparent Print Data command provides printing access to those characters which the printer normally defines as unprintable. These characters include decimal character codes 0, 7-15, and 27.

$^{E}_{C}$ **& p # X** [Transparent Print Data]

\# =Number of bytes of transparent print data.

Default = N/A

Range = 0 - 32767

Each transparent print data byte is interpreted as a single character code. The appropriate character is printed if one exists; otherwise, a Space is processed. For example, control codes such as LF, CR, FF are treated as print data while in Transparent Print Data mode.

Example

Assuming the currently selected symbol set is PC-8, send the following to print musical notes (decimal code 14):

$^{E}_{C}$**&p1X** *[ASCII 14]*

The brackets "*[ ]*" are provided for clarity and are not part of the command sequence.

Note

In the ASCII symbol set, decimal 14 is the Shift Out control code (no printable character exists), however, in the PC-8 symbol set, decimal code 14 is also the musical notes character (printable from transparency mode). Refer to Appendix A for character codes for the various symbol sets.

Underline Command

The Underline command controls automatic text underlining.

$^{E}{}_{C}$ **& d # D -** Enable underline

\# = 0 - Fixed position
3 - Floating position

Default = 0

Range = 0, 3 (values outside range are ignored)

$^{E}{}_{C}$ **& d @ -** Disable underline

Once underlining is enabled, any positive horizontal movement causes an underline to be drawn. Positive horizontal movement includes the printing of text and positive horizontal cursor motion.

When fixed position underlining is enabled, the underline is drawn five dots below the baseline and is three dots thick. (The baseline is the dot row on which all of the characters in a given line appear to stand, see Chapter 11.) When floating position underline is enabled, the underline position is determined by the greatest underline distance below the baseline of all of the fonts printed on the current line. (The underline distance for a font is defined in the font header, see Chapter 11.)

Note The underline and the underscore character may not necessarily be aligned or be the same thickness.

9 Font Management

Introduction

Font management provides mechanisms for manipulating soft fonts. It provides the means for controlling which soft fonts are saved in user memory (RAM) or deleted. This is accomplished by assigning a font as either temporary or permanent, or deleting a soft font. In addition, font management includes the command for assigning ID numbers to RAM fonts. It also provides a mechanism for copying ROM fonts (internal, cartridge, or SIMM) to RAM for the purpose of assigning ID numbers.

Downloading Soft Fonts

The process of transferring soft fonts from a host computer to the printer's user memory (RAM) is called downloading. Designate a unique identification (ID) number prior to the download of a font. This number is then associated with the soft font. This number is assigned using the Font ID command, described later in this chapter. Subsequent manipulation of the soft font is accomplished using the font's ID number. If a font is already associated with this ID number in the printer, the existing font is deleted during the download.

Several commands are required to define a font before downloading it to the printer. These commands are described in detail in Chapter 11. Hewlett-Packard font files include the necessary commands that define the symbols of a font. Assigning a font ID number and then copying the font file to the printer downloads the font. Scalable fonts and typefaces may be prepared by font management software such as HP's Type Director. Once prepared, scalable fonts are downloaded in much the same manner as bitmap fonts.

Once downloaded, a soft font occupies a portion of user memory (RAM). The number of soft fonts that can be stored in user memory is limited only by the amount of available user memory.

Temporary vs. Permanent Fonts

Once downloaded, a font is automatically designated as temporary. A temporary soft font is deleted from user memory during a printer reset or when a Typeface List, a Font Printout or a self-test is performed from the printer's control panel. A soft font can be designated as permanent to prevent the printer from deleting it during a printer reset. A soft font is designated as temporary or permanent by referencing its ID number and using the Font Control command (refer to "Font Control Command" later in this chapter).

Note Both temporary and permanent fonts are deleted from user memory whenever the printer's power is turned off.

Switching printer languages ("personalities"), changing resolution, or changing the Page Protection setting also deletes temporary and permanent fonts from user memory in some printers (refer to the appropriate printer *User's Manual* for specifics).

An existing font is deleted when a new font with the same ID number is downloaded. The new font replaces the existing font (whether temporary or permanent).

Deleting Fonts

There are several mechanisms provided by PCL font management that delete soft fonts from user memory. These include commands to delete all soft fonts, all temporary soft fonts, or an individual soft font by reference to its font ID number (refer to the Font ID and the Font Control commands described on the following pages).

Font ID Command

The Font ID command is used to specify an ID number for use in subsequent font management commands. The ID number of a font can be used to select the font for printing (refer to “Font Selection by ID” in Chapter 8).

$^{E}{}_{C}$ ***c#D**

\#   =   ID number

Default **=** 0

Range **=** 0 - 32767

The font ID number is used during subsequent soft font downloads, selections or deletions.

The factory default font ID is 0 (if no Font ID command is sent, an ID of 0 is assigned).

Note The font number assigned by the printer and used from the printer’s control panel is not the same as the ID number assigned using the Font ID Command.

Example

To specify a font ID number of 1, send:

$^{E}{}_{C}$***c1D**

Font Control Command

The Font Control command provides mechanisms for manipulating soft fonts.

$^{E}{}_{C}$ *** c # F**

\# = 0 - Delete all soft fonts
1 - Delete all temporary soft fonts
2 - Delete soft font (last ID specified)
3 - Delete Character Code (last ID and Character Code specified)
4 - Make soft font temporary (last ID specified)
5 - Make soft font permanent (last ID specified)
6 - Copy/Assign current invoked font as temporary (last ID specified)

Default = N/A

Range = 0 - 6 (values outside range are ignored)

Note If the primary or secondary font is deleted, a new primary or secondary font is selected automatically from the remaining fonts.

Examples

To remove all soft fonts from user memory, send:

$^{E}{}_{C}$ ***c0F**

To remove only those soft fonts that are temporary, send:

$^{E}{}_{C}$ ***c1F**

To delete the soft font with an ID of 1, send:

$^{E}{}_{C}$ ***c1d2F**

To delete the character “p” (112 decimal) in a bitmap or bound scalable font with an ID of 1, send:

$^{E}_{C}$ ***c1d112e3F**

(A space is printed in place of the deleted character. Also, the $^{E}_{C}$***c#E** Character Code command used in the above sequence “...112e...,” is described in Chapter 11.)

To make the soft font with an ID of 2 temporary, send:

$^{E}_{C}$ ***c2d4F**

To make the soft font with an ID of 2 permanent, send:

$^{E}_{C}$ ***c2d5F**

To make a copy of the currently invoked (selected) font, with an ID of 9, send:

$^{E}_{C}$ ***c9d6F**

The Copy/Assign font control feature can be used to copy either ROM or RAM fonts into RAM assigning them ID numbers.

Note

When the currently selected font is a scalable TrueType ROM font, $^{E}_{C}$*c#d6F **assigns** a font ID number, but makes **no copy** of the font in RAM. Any attempts to download or delete characters within the font are ignored. An attempt to delete the font merely results in the loss of the ID number.

Font Management Example

This example illustrates several typical font management operations. It assumes a bitmap soft font is stored and available on an MS-DOS based hard disk.

1 Set the font ID number to 2:

$^{E}{}_{C}$***c2D**

2 Download a soft font file using the MS-DOS COPY command with the /B option:

COPY /B *filename* PRN

Note that the soft font is associated with font ID 2.

3 Make the soft font permanent to prevent its deletion during a printer reset:

$^{E}{}_{C}$***c5F**

4 Designate the permanent soft font as primary:

$^{E}{}_{C}$**(2X**

Unbound Scalable Fonts

Prior to introduction of the HP LaserJet IIIP printer, a downloaded scalable font was restricted to a single symbol set. Now scalable fonts with no symbol set affiliation can be downloaded. These new fonts are called unbound fonts.

To download unbound fonts, the “PCL Font Header for Intellifont Unbound Scalable Fonts” or the “Format 15 Font Header for Scalable Fonts” (TrueType) must be used (see Chapter 11).

Bound and Unbound Fonts

The terms “bound” and “unbound” refer to the symbol set capacity of a font. A bound font identifies a font which is restricted (bound) to a single symbol set. An unbound font (or unbound *typeface*) indicates the capacity to be bound to a set of symbols selected from a complementary **symbol index** (such as the Master Symbol List (**MSL**), or the **Unicode** symbol index).

Font Selection and Unbound Fonts

When a font is requested for printing, the printer selects a font which most closely matches the current font selection characteristics (symbol set, spacing, pitch, height, style, stroke weight, and typeface). Refer to “Summary of Font Selection by Characteristics” in Chapter 8 for detailed font characteristic selection information.

Since symbol set is the highest font selection priority and typeface is the lowest, the printer searches for the symbol set first. A list of all fonts that match the requested symbol set is made. This includes bitmap, bound, and unbound fonts. Since bitmap and bound scalable fonts contain only one symbol set, they can be easily identified. However, determining which unbound fonts match a symbol set is more complex. Symbol set compatibility for unbound fonts is determined by identification of groups of symbols referred to as symbol collections.

Symbol Collections

The symbols in an unbound font (typeface) can be divided into symbol collections. These symbol collections identify the symbols according to some language basis or special application usage. Some symbol collections include: Basic Latin, East European, Turkish, Math, Semi-Graphic, and Dingbats. If a symbol collection is included in an unbound font, all of the symbols of that collection are included.

Note See Appendix D in the *PCL 5 Comparison Guide* to identify symbols in the various collections.

The symbols within a symbol collection do not change from one unbound font to the next. For example, the Basic Latin collection always contains the same symbols. Different fonts may contain different symbol collections. For example, the internal Univers typeface contains the Latin, Math, and Semi-Graphic collections (these collections contain all the symbols required for the 35 symbol sets that Univers supports). The ITC Zapf Dingbats typeface, on the other hand, contains only the Dingbats collection, which includes all the symbols required for the five supported symbol sets.

When searching unbound scalable fonts (during font selection) for those that match the requested symbol set, the printer actually searches for symbol collections. To identify symbol collections which meet the needs of the requested symbol set, the printer uses two numbers: the **Character Requirements** number and the **Character Complement** number.

Character Complement Numbers

The “Intellifont Unbound Scalable Font Header” (header) includes a 64 bit field (bytes 78-85) which contains the Character Complement number. For TrueType fonts, in the “Format 15 Font Header for Scalable Fonts” (unbound), the Character Complement number is included in the accompanying “Segmented Font Data” section of the header.

The Character Complement number identifies the symbol collections in the font. Each bit in this field corresponds to a symbol collection (not all bits are currently defined; refer to Appendix D in the *PCL 5 Comparison Guide*).

Intellifont example: If bits 63 and 34 are cleared (set to zero) it indicates that the unbound font contains the Basic Latin (bit 63) and Math (bit 34) symbol collections and that the character index is in HP’s MSL numbers (bit 0).

TrueType example: If bits 31, 30, and 0 are cleared (set to zero), it indicates that the unbound font contains ASCII, Latin 1 extensions and is based on Unicode numbers.

Character Requirements Number

The other number the printer uses to determine symbol set compatibility, the Character Requirements number, is provided as part of the information contained in the symbol set. The Character Requirements number is a 64-bit number analogous to the Character Complement number; however, it identifies the symbol collections needed by the symbol set.

Intellifont example: If a symbol set based on HP’s MSL numbers requires one or more characters from the standard Latin collection and some of the characters from the Math collection then bits 63 and 34 are set to one (refer to Appendix D in the *PCL 5 Comparison Guide* for information regarding the various symbol collections).

TrueType example: If a symbol set based on Unicode numbers requires one or more characters from the standard ASCII collection and some of the characters from the Latin 5 collection, then bits 31, 28 and 0 are set to one (refer to Appendix D in the *PCL 5 Comparison Guide* for information regarding the various symbol collections).

As stated above, to determine which unbound scalable fonts contain the symbols for the specific symbol set, the printer must identify those unbound fonts that contain the symbol collections of the requested symbol set. To do this, the printer accesses the Character Requirements number for the requested symbol set. If, for example, the Roman-8 symbol set was requested, the printer would access the Character Requirements number from the Roman-8 symbol set information in the printer. This number is then compared with the Character Complement number of each unbound font in the printer. If any matches are found, those unbound fonts are included in the list of potential fonts for selection.

Final Font Selection

After the process above is complete, the printer contains a list of all fonts (bitmap, bound, and unbound) which support the requested symbol set. (If no fonts are found for the specified symbol set, Roman-8 is used. If more than one font remains, the printer continues comparing font selection characteristics, eliminating fonts, until only one remains. Whenever only one font remains, it is selected for printing.

Symbol Set Mapping Table

The printer receives character codes in the range 0-255 which, depending on the selected symbol set, identify the symbols to print or control codes to execute.

There are hundreds of symbols available in unbound fonts in HP LaserJet printers, more than can be identified by the character code range (0 - 255). A list of these symbols is provided in the **symbol indexes**, such as the Master Symbol List (MSL) and the Unicode list In Appendix D of the *PCL 5 Comparison Guide*). Each symbol in the list is identified by a unique MSL or Unicode number. Symbols in unbound fonts are identified by this number.

Since the printer identifies symbols by their symbol index number (range from 0 to 65535), but receives character codes (range 0-255), a relation must be made between the character codes and the larger range of symbol index numbers. This relation is defined by the symbol set mapping table.

The printer contains a symbol set mapping table for each available symbol set. These tables list the character code range and corresponding list of symbol index numbers. Using this mapping the printer identifies which indexed character is printed for the character code in the current symbol set.

A partial symbol set mapping table is shown in Table 9-1 for the Roman-8 symbol set.

Table 9-1 Roman-8 Symbol Index Mapping

Character Code	MSL Index (decimal)	Unicode Index (hexadecimal)
32	0	
33	1	0021
34	2	0022
35	3	0023
36	4	0024
37	5	0025
38	6	0026
39	8	2019
40	9	0028
41	10	0029
⋮	⋮	⋮
252	189	25a0
253	190	00bb
254	191	00b1

Printing a Character

When an unbound font is selected for printing and a character code is received, the printer accesses the requested symbol set mapping table to identify the MSL or Unicode number. For example, if the Roman-8 symbol set is selected and the printer receives character code 254, the printer accesses the Roman-8 symbol set mapping table (Table 9-1). In the Roman-8 mapping table, character code 254 is mapped to MSL number 191 or Unicode number 00b1 (plus-over-minus symbol). Thus, to print character code 254, the printer searches the selected unbound font for the correct MSL or Unicode number and prints that character.

10 User-Defined Symbol Sets

Introduction

User-defined symbol sets are used with unbound scalable fonts. Three new commands provide for the implementation of user-defined symbol sets:

Symbol Set ID Code - $^{E}{}_{C}$***c#R**

Define Symbol Set - $^{E}{}_{C}$**(f#W [symbol set data]**

Symbol Set Control - $^{E}{}_{C}$***c#S**

To define a symbol set, you must first designate a symbol set ID code. Next, use the Define Symbol Set command to download the list of characters (character codes and related symbol index numbers) for the symbol set. Once this is accomplished, you may select the symbol set for printing in the same manner as any symbol set using the symbol set selection sequence: $^{E}{}_{C}$**(ID**

Once a user-defined symbol set is downloaded, the Symbol Set Control command can be used to assign symbol sets as either temporary or permanent and to delete them.

Symbol Set ID Code Command

The Symbol Set ID Code command assigns a symbol set ID code to a user-defined symbol set. This ID code is used by the Symbol Set Control command for symbol set management.

The ID code corresponds to the symbol set ID selection value which is used to identify the symbol set during font selection.

$^{E}{}_{C}$ *** c # R**

\# = Symbol Set ID Code (decimal)

Default = 0

Range = 0 - 32,767 (larger values are outside the legal range)

When downloading a symbol set, the symbol set ID value must match the Encoded Symbol Set Designator field in the user-defined symbol set header.

As mentioned above, the symbol set ID code is related to a symbol set ID value. The relationship between the ID code and the symbol set ID selection value is shown by the following formula:

Symbol Set ID code = (# * 32) + (ID - 64)

where: "#" represents the number portion of the ID selection value which may range from 0-1023; and, "ID" represents the ordinal (decimal) value of the ID character. (Symbol set ID selection values consist of a number and a letter, such as 8U for Roman-8 or 7J for DeskTop etc.)

For example:

Assume the ID selection value selected for this symbol set is 17Q, then:

(17 * 32) + (81 - 64) = 561

The symbol set ID code is 561.

When selecting an ID code, select one which is not being used currently. (If an ID code is selected which is already being used in the printer, that symbol set is redefined.) The first step in selecting an ID code is to determine an ID selection value. Since it may be difficult to determine which ID selection values are being used, it is best to select one which has not been assigned. Refer to Appendix C in the *PCL 5 Comparison Guide* to select a number/letter (ID selection value) combination which has not been assigned.

After the ID selection value has been determined, use the conversion formula (shown above) to convert the ID value to an ID code.

To create a user-defined symbol set:

1 Identify the symbols (symbol index numbers) for the symbol set from the MSL or Unicode list in Appendix D of the *PCL 5 Comparison Guide*. List them in the appropriate character code order.

2 Identify the symbol collections (and Character Requirement bits) that contain the symbols (Appendix D of the *PCL 5 Comparison Guide*).

3 Identify the Define Symbol Set command header information.

4 Identify a symbol set ID selection value and convert it into its symbol set ID code.

5 Designate the ID code using the Symbol Set ID Code command.

6 Download the symbol set header data and MSL or Unicode numbers using the Define Symbol Set command.

To print using the symbol set, it must be selected using the Select Symbol Set command - $^{E}{}_{C}$**(ID**, where **ID** is the symbol set ID selection value used to calculate the symbol set ID code.

Define Symbol Set

This command defines the characters and character mapping for a user-defined symbol set.

$^{E}{}_{C}$ (f # W [symbol set definition data]

= Number of bytes in symbol set definition

Default = n/a
Range = 0 - 32767

If a user-defined symbol set is already present in the printer and a new, valid symbol set is downloaded with the same Symbol Set ID Code, then the old symbol set is deleted. If an internal symbol set with the same ID code exists, it is overridden by the new symbol set. If the symbol set definition is invalid, the command is ignored. If there is insufficient memory to create the symbol set, the symbol set is discarded.

Note Send the Symbol Set ID Code command prior to the Define Symbol Set command to assign an ID code for the user-defined symbol set. If the Symbol Set ID Code command is not sent, the last code sent is used. If none have been sent, then the default (0) is assigned.

The data format for the user-defined symbol set is shown in Table 10-1.

Table 10-1 User-Defined Symbol Set Defintion Format

Byte	15 - MSB8	7LSB - 0
0	Header Size (18)	
2	Encoded Symbol Set Designator	
4	Format	Symbol Set Type
6	First Code	
8	Last Code	
10	Character Requirements	
Hdr Size	Symbol Map [Last Code — First Code + 1] ⋮	

The following abbreviations are used to define the data type of each field in the symbol set header:

Font Header Field Data Type Notation		
(B)	: Boolean	(0, 1)
(UB)	: Unsigned Byte	(0 . . 255)
(SB)	: Signed Byte	(-128 . . 127)
(UI)	: Unsigned Integer	(0 . . 65535)
(SI)	: Signed Integer	(-32768 . . 32767)
(ULI)	: Unsigned Long Integer	(0 . . 2^{32}-1)
(SLI)	: Signed Long Integer	(-2^{31} . . 2^{31}-1)
(ASCxx)	: ASCII string	array (0 . . xx-1) of characters

Header Size (UI)

Set the header size to the size of the header data — the number of bytes from Header Size (byte 0) to the last byte just before the beginning of the Symbol Map data bytes. This value is 18 or greater.

Encoded Symbol Set Designator (UI)

This field must match the ID code in the Symbol Set ID code command.

This field contains the symbol set ID code. The symbol set ID code is calculated from a symbol set ID selection value using the following formula:

Symbol Set ID Code = (# * 32) + (ID - 64)

where # is the decimal number (0 to 1023) and ID is the ASCII character code of the letter.

For example:

Assume the ID selection value selected for this symbol set is 17Q, then:

(17 * 32) + (81 - 64) = 561

The symbol set ID code is 561.

Format (UB)

Set this field to 1 for MSL (Intellifont) or 3 for Unicode (TrueType).

Unrecognized values cause the symbol set definition to be ignored.

Symbol Set Type (UB)

This field defines the printable and unprintable codes for the symbol set.

Bit Field	Designated Use
0	7-bit, 32-127 are printable.
1	8-bit, 32-127 and 160-255 are printable.
2	8-bit, 0 - 255 character codes are printable, however, to print codes 0, 7-15, and 27, the printer must be in transparency mode.

First Code (UI)

Specifies the first character code in the set.

Last Code (UI)

The Last Code specifies the last character code in the set.

Together, the First Code through the Last Code identify the range of character codes which map to the symbol index numbers (characters) in the Symbol Map field.

Character Requirements (Array of UB) and character requirement

This 8-byte field works in conjunction with the Character Complement field in the header of a type 10 or 11 (unbound) font to determine the compatibility of a symbol set with an unbound font. These two fields identify the unbound fonts in the printer which contain the symbol collections required to build a symbol set. Refer to “Unbound Scalable Fonts” in Chapter 9, for a description of symbol collections and unbound fonts.

Each bit in the field represents a specific collection. Setting a bit to 1 indicates that collection is required; setting the bit to 0 indicates that collection is not required. (Bit 63 refers to the most significant bit of the first byte, and bit 0 refers to the least significant bit of the eight byte field.) The bit representations for the collections are shown below. (The symbols for each collection are shown in Appendix D of the *PCL 5 Comparison Guide*.)

MSL Symbol Index Character Requirements		
Bit	**Value**	**Designated Use**
63	1	Basic Latin required (such as ISO 8859/1 Latin 1)
	0	Basic Latin not required
62	1	East European Latin required (such as ISO 8859/2 Latin 2).
	0	East European Latin not required
61	1	Turkish required (such as ISO 8859/9 Latin 5)
	0	Turkish not required
34	1	Math required (such as Math-8)
	0	Math not required
33	1	Semi-graphic required (such as PC-8 D/N)
	0	Semi-graphic not required
32	1	Dingbats required (such as ITC Zapf Dingbats series 100, series 200, etc.)
	0	Dingbats not required
2,1,0	000	MSL Symbol Index

Unicode Symbol Index Character Requirements Bits (numbers/values)		
Bit	**Value**	**Designated Use**
31	1	ASCII required (such as ISO 6 ASCII)
	0	ASCII not required.
30	1	West Europe extensions required (such as ISO 69 French).
	0	West Europe extensions not required.
29	1	East Europe extensions required (such as ISO 8859/2 Latin 2).
	0	East Europe extensions not required.
28	1	Turkish extensions required (such as ISO 8859/9 Latin 5).
	0	Turkish extensions not required.
27	1	Desktop Publishing extensions required (such as Windows 3.1).
	0	Desktop Publishing extensions not required.
26	1	Accent extensions required (such as ISO 8859/1 Latin 1).
	0	Accent extensions not required.
25	1	PCL extensions required (such as Roman-8).
	0	PCL extensions not required.
24	1	Macintosh extensions required (such as MC Text).
	0	Macintosh extensions not required.
23	1	PostScript extensions required (such as PS Text).
	0	PostScript extensions not required.
22	1	Code Page extensions required (such as PC-8).
	0	Code Page extensions not required.
2,1,0	001	Unicode Symbol Index

Examples of values for the field include:

Bit Field	Designated Use
Value (Hex)	**Meaning**
0000000000000000	Default requirement (MSL); symbol set can be used with any typeface indexed bu MSL.
8000000000000000	Symbol set (MSL) requires only the Basic Latin Symbol Collection (such as Roman-8)
0000000100000000	Symbol set (MSL) requires only the Dingbat Collection.
0000000000000001	Default requirement (Unicode); symbol set can be used with any typeface indexed bu Unicode.
00000000A0000001	Symbol set (Unicode) requires the ASCII and East Europe Collections (such as ISO 8859/2).
0000000088000001	Symbol set (Unicode) requires the ASCII and Desktop Publishing Collections (such as Ventura US).

Symbol Map (Array of UI)

The symbol map contains a list of symbol index numbers. This list identifies symbols for the symbol set. (Refer to Appendix D in the *PCL 5 Comparison Guide* for a MSL and Unicode symbol indexes.) The symbol map pairs (maps) a character code to a symbol index number. The range of character code numbers (paired with symbol index numbers) is the range from the First Code through the Last Code fields in the header. The first symbol index number in the Symbol Map field is mapped to the character code whose value is that of the First Code field; the second symbol index number is mapped to the “First Code + 1” character code; the third symbol index number is mapped to the “First Code + 2,” etc., through the last symbol index number, which is mapped to the value in the Last Code field. The number of symbol index characters in the array must match the number of character codes in the range, First Code through Last Code.

If no printable symbol (symbol index number) is associated with a given character code (as with codes 128 through 160 of Roman-8), the corresponding entry in the Symbol Map should be 65535 (FFFF Hex).

Symbol Set Control Command

This command provides a means for making user-defined symbol sets permanent or temporary, and for deleting them.

$^{E}{}_{C}$ * c # S

\# = 0 -Delete all temporary and permanent user-defined symbol sets.
1 - Delete all temporary user-defined symbol sets.
2 - Delete current user-defined symbol set (last symbol set ID code specified).
4 - Make current user-defined symbol set temporary.
5 - Make current user-defined symbol set permanent.

Default = n/a
Range = 0-2, 4, 5 (other values ignored)

Downloaded symbol sets default to temporary.

Internal symbol sets cannot be deleted or made temporary.

User-Defined Symbol Set Examples

The following two examples illustrate the concept of user-defined symbol sets. They create symbol sets for PC-8 in MSL and Unicode symbol indexes. The necessary escape sequences are shown in each example.

Unicode Symbol Index Example

Symbol Map Data:	
${}^{E}{}_{C}$*c341R	PCL Symbol Set #IDs: 10U
${}^{E}{}_{C}$(f526W	Symbol Set 526 bytes in length
00 12	Header Size 18 bytes
01 55	ID code 341 decimal: 10U
03	Format 3 (Unicode Symbol Index)
02	Font Type 2
00 01	First code = 1
00 fe	Last code = 254
00 00 00 00 c0 40 00 01	ASCII, Latin 1, and PC Characters required
26 3a (character code 1)	Open Happy Face
26 3b (character code 2)	Solid Happy Face
26 65 (character code 3)	Solid Heart, Card Suit
26 40 (character code 4)	Solid Diamond, Card Suit
26 63 (character code 5)	Solid Spade, Card Suit
⋮	⋮
25 bc (character code 31	Down Solid Arrowhead
ff ff (character code 32)	Space Code (no character)
00 21 (character code 33)	Exclamation Mark
⋮	⋮

Symbol Map Data: ***(continued)***	
00 41 (character code 65)	Uppercase A
⋮	⋮
00 61 (character code 97)	Lowercase A
⋮	⋮
20 7f (character code 252)	Superior Lowercase N
00 b2 (character code 253)	Superior Numeral 2
25 a0 (character code 254)	Small Solid Square Box
$^{E}{}_{C}$*c341r5S	PCL Symbol Set #ID: 10U, Make this symbol set permanent.

MSL Symbol Index Example

Symbol Map Data:	
$^{E}{}_{C}$*c341R	PCL Symbol Set #ID: 10U
$^{E}{}_{C}$(f528W	Symbol Set 528 bytes in length
00 12	Header Size 18 bytes
01 55	ID code 341 decimal: 10U
01	Format 1 (MSL Symbol Index)
02	Font Type 2
00 01	First code = 1
00 ff	Last code = 255
80 00 00 02 00 00 00 00	Basic Latin and PC Characters required
00 cb (character code 1)	Open Happy Face
00 cc (character code 2)	Solid Happy Face
00 cd (character code 3)	Solid Heart, Card Suit
00 ce (character code 4)	Solid Diamond, Card Suit

Symbol Map Data: ***(continued)***	
00 cf (character code 5)	Solid Spade, Card Suit
⋮	⋮
00 e7 (character code 31)	Down Solid Arrowhead
00 00 (character code 32)	Space Code
00 01 (character code 33)	Exclamation Mark
⋮	⋮
00 22 (character code 65)	Uppercase A
⋮	⋮
00 43 (character code 97)	Lowercase A
⋮	⋮
01 4c (character code 252)	Superior Lowercase N
00 c5 (character code 253)	Superior Numeral 2
01 31 (character code 254)	Small Solid Square Box
00 00 (character code 255)	No-Break Space
$^{E}{}_{C}$*c341r5S	PCL Symbol Set #ID: 10U,

11 Soft Font Creation

Introduction

A font that is downloaded (transferred) from a computer to a printer is called a soft font. A PCL soft font contains a **font header** and a set of **character definitions**. The font header and character definitions contain all the information needed to format a font for use in the HP LaserJet printers.

Every PCL font header begins with a font descriptor, which identifies the basic characteristics common to all characters of a font, such as font type, baseline position, character cell width and height, character orientation, symbol set, etc.

Every PCL character definition contains a character descriptor and a body of character data. Furthermore, the character definition always consists of one or more character data blocks. Each character data block begins with its own character data block header.

The character descriptor is a block of data that identifies the characteristics for a specific character, such as its position, and the cursor position after printing. The character data which follows defines the shape of the character.

This chapter describes the font header and character definition formats for PCL Bitmap fonts, Intellifont scalable and TrueType scalable fonts. By formatting a font consistent with the header format requirements, a user may download this information (the font) to the printer using the Font Header command and the Character Descriptor/Data command. One additional command, the Character Code command, required to identify the ASCII character code assigned to each character, is also described in this chapter.

The definition of a font with a quantity *n* characters would appear as shown below.

Table 11-1

Font ID Command
Font Header
Character $Code_1$
Character $Descriptor_1$
Character $Data_1$
Character $Code_2$
Character $Descriptor_2$
Character $Data_2$
⋮
Character $Code_n$
Character $Descriptor_n$
Character $Data_n$

Font Classifications

There are three basic classifications of fonts accepted by the HP LaserJet printer: **PCL bitmap**, **Intellifont scalable**, and **TrueType scalable**. Several different font header and character descriptor formats are available for the different font classifications. All are presented in this chapter.

Note Not all font classifications are supported in all HP LaserJet Family printers. Refer to the *PCL 5 Comparison Guide* or the printer *User's Manual* for specific information.

With the information provided in the section for bitmap fonts, it is possible to format a PCL bitmap character/font for the printer. However, to format an Intellifont or TrueType scalable font, additional information is required.

Intellifont scalable fonts are formatted to use Agfa Scaling Technology. Intellifont scalable fonts are described in detail in the document, *Intellifont Scalable Typeface Format*, available from Agfa Division, Miles Inc. (Refer to *Related Documents*, located in the front of this manual, for information on how to obtain this document.)

TrueType scalable fonts are described in detail in the document, *True Type Font Files*. (Refer to *Related Documents*, located in the front of this manual, for information on how to obtain this document.)

Note The documents, *Intellifont Scalable Typeface Format* and *True Type Font Files* do not contain descriptions of scalable PCL fonts. Instead, they contain descriptions of files from which PCL fonts can be built.

Coordinate System

Both bitmap and scalable characters are designed in an area referred to as a *cell* or *window,* and each has its own coordinate system and set of units.

Bitmap Fonts

Characters of a bitmap font are designed within a rectangular area referred to as a cell. The bitmap character cell is illustrated in Figure 11-3, Figure 11-6, and Figure 11-7. The physical coordinate system is defined in terms of the directions of raster scan (X) and paper motion (Y), as illustrated in the following figure.

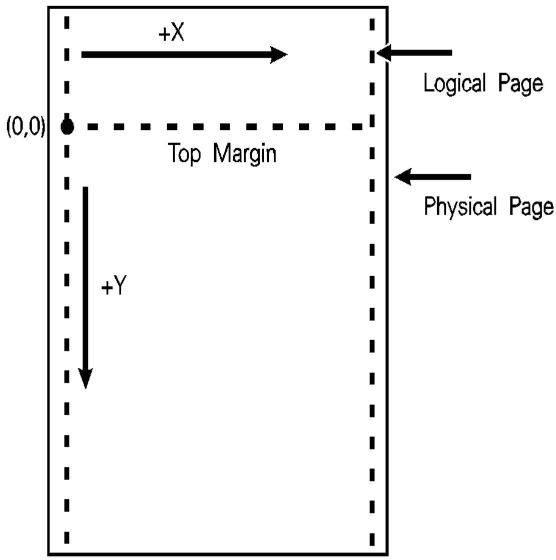


Figure 11-1 Bitmap Physical Coordinate System

Note The LaserJet IID, IIP, 2000, and all LaserJet PCL 5 printers rotate fonts to match the paper's physical coordinate system.

Intellifont Scalable Fonts

Characters of an Intellifont scalable font are designed within a rectangular area known as the Agfa Design Window (Figure 11-2). The units of this coordinate system are .01mm square.

The master font design size is 250 points (a CG point=.01383 inches). There are 8782 units per Em at the Master Font Size.

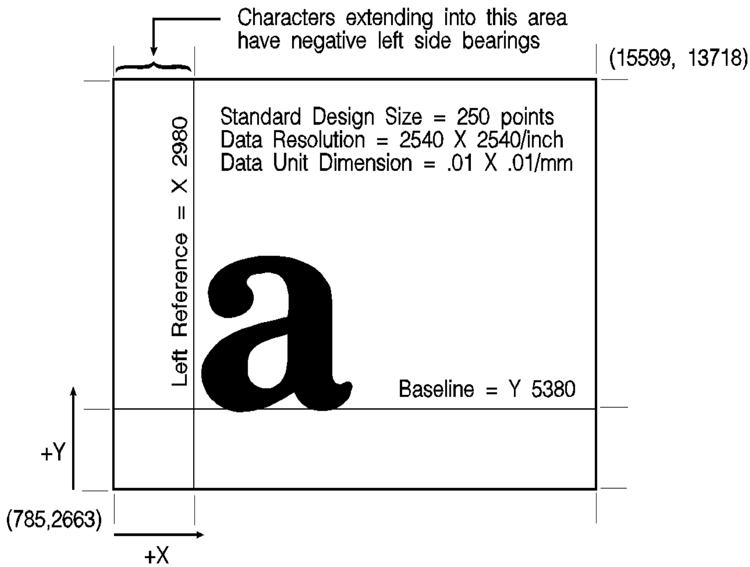


Figure 11-2 Agfa Design Window

TrueType Scalable Fonts

TrueType scalable font character coordinates are typically based on a system of 2048 units per Em. The baseline is defined by y-coordinate=0. The left reference is usually defined by x-coordinate=0 (although there is some variation among fonts). To determine the left reference line for an individual TrueType character, subtract the LSB value found in the hmtx table from the xMin value found in the glyf table. See *True Type Font Files* for more information.

Font Header Command

The Font Header command is used to download font header data to the printer.

$^{E}{}_{C}$ **) s # W** [font header data]

Default = 0
Range = 0 - 32767

The value field (#) identifies the number of bytes in the font header. The font header fields are described under *Font Header Format*, later in this chapter.

Note Two examples for downloading a Font Header are provided under *Font Header Examples*, after the *Font Header Format* section, later in this chapter.

Font Header Format

The **font header** describes those characteristics of a font which are common to all its characters. Five font headers are included here:

- **Format 0 Font Header for PCL Bitmapped Fonts** - This font header, shown in Table 11-2, is not recommended for LaserJet 4 and later printers. It is included in this manual to maintain backward-compatibility with earlier versions of PCL.
- **Format 20 Font Header for Resolution-Specified Bitmapped Fonts** - This header replaces the previous bitmap header, and has the added capability to specify a font's resolution. This font header is shown in Table 11-3.
- **Format 10 Font Header for Intellifont Bound Scalable Fonts** - For creating Intellifont scalable fonts which are restricted (bound) to a single symbol set. This font header is shown in .
- **Format 11 Font Header for Intellifont Unbound Scalable Fonts** - For creating Intellifont scalable fonts which are not bound to a single symbol set, but are instead composed of a range of compatible symbol collections. This font header is shown in .
- **Format 15 TrueType Scalable Font Header** - This new font header supports TrueType scalable fonts (bound or unbound). This font header is shown in Table 11-6.

Note Use the Font ID command to designate a unique ID number prior to the download of a font header. If an existing font is already associated with this ID, the existing font is deleted upon the download of the font header. Unless otherwise specified, inappropriate values in a font header field invalidates the font download process; a font is not created and the associated font data is discarded.

Intellifont scalable font formatting also requires the *Intellifont Scalable Typeface Format* document, which supplements the information provided here. For information on how to obtain this document, refer to "Related Documents" in the front of this manual.

The figures that follow illustrate the font header formats for the various font classifications. The individual fields for the font headers are described following the figures.

Notes Although some LaserJet printer models do not use all of the data in the font header and thus ignore many of the fields, a font creator should use valid values in all of the font header fields. This ensures font compatibility across the LaserJet printer family and with future printers, which may use these fields.

Those font header fields identified as "reserved" should be set to zero.

Table 11-2 Format 0 Font Header (for PCL Bitmapped Fonts)

Byte	15 (MSB) 8	7 (LSB) 0
0	Font Descriptor Size (64)	
2	Header Format (0)	Font Type
4	Style MSB	Reserved
6	Baseline Position	
8	Cell Width	
10	Cell Height	
12	Orientation	Spacing
14	Symbol Set	

Table 11-2 Format 0 Font Header (for PCL Bitmapped Fonts)

16	Pitch (Default HMI)	
18	Height	
20	x-Height	
22	Width Type	Style LSB
24	Stroke Weight	Typeface LSB
26	Typeface MSB	Serif Style
28	Quality	Placement
30	Underline Position (Distance)	Underline Thickness (Height)
32	Text Height	
34	Text Width	
36	First Code	
38	Last Code	
40	Pitch Extended	Height Extended
42	Cap Height	
44-47	Font Number ⋮	
48-63	Font Name ⋮	
64	Copyright (optional)	

Table 11-3 Format 20 Font Header (for Resolution-Specified Bitmapped)

Byte	15 (MSB) 8	7 (LSB) 0
0	Font Descriptor Size (68)	
2	Header Format (20)	Font Type
4	Style MSB	Reserved
6	Baseline Position	

Table 11-3 Format 20 Font Header (for Resolution-Specified Bitmapped) (continued)

8	Cell Width	
10	Cell Height	
12	Orientation	Spacing
14	Symbol Set	
16	Pitch (Default HMI)	
18	Height	
20	x-Height	
22	Width Type	Style LSB
24	Stroke Weight	Typeface LSB
26	Typeface MSB	Serif Style
28	Quality	Placement
30	Underline Position (Distance)	Underline Thickness (Height)
32	Text Height	
34	Text Width	
36	First Code	
38	Last Code	
40	Pitch Extended	Height Extended
42	Cap Height	
44-47	Font Number ⋮	
48-63	Font Name ⋮	
64	X Resolution	
66	Y Resolution	
n	Copyright (optional) ⋮	

Table 11-4 Format 10 Font Header (for Intellifont Bound Scalable)

Byte	15 (MSB) 8	7 (LSB) 0
0	Font Descriptor Size (minimum 80)	
2	Header Format (10)	Font Type
4	Style MSB	Reserved
6	Baseline Position	
8	Cell Width	
10	Cell Height	
12	Orientation	Spacing
14	Symbol Set	
16	Pitch (default HMI)	
18	Height	
20	x-Height	
22	Width Type	Style LSB
24	Stroke Weight	Typeface LSB
26	Typeface MSB	Serif Style
28	Quality	Placement
30	Underline Position (Distance)	Underline Thickness (Height)
32	Text Height	
34	Text Width	
36	First Code	
38	Last Code	
40	Pitch Extended	Height Extended
42	Cap Height	
44-47	Font Number ⋮	

Table 11-4 Format 10 Font Header (for Intellifont Bound Scalable) (continued)

48-63	Font Name ⋮	
64	Scale Factor	
66	X Resolution	
68	Y Resolution	
70	Master Underline Position	
72	Master Underline Thickness (Height)	
74	OR Threshold	
76	Global Italic Angle	
Desc. Size-2	Global Intellifont Data Size	
80	Global Intellifont Data ⋮	
n	Copyright (optional) ⋮	
	Reserved (0)	Checksum

Table 11-5 Format 11 Font Header (for Intellifont Unbound Scalable Fonts)

Byte	**15 (MSB) 8**	**7 (LSB) 0**
0	Font Descriptor Size (minimum 88)	
2	Header Format (11)	Font Type (10)
4	Style MSB	Reserved
6	Baseline Position	
8	Cell Width	
10	Cell Height	
12	Orientation	Spacing
14	Symbol Set	

Table 11-5 Format 11 Font Header (for Intellifont Unbound Scalable Fonts) (continued)

16	Pitch (default HMI)	
18	Height	
20	x-Height	
22	Width Type	Style LSB
24	Stroke Weight	Typeface LSB
26	Typeface MSB	Serif Style
28	Quality	Placement
30	Underline Position (Distance)	Underline Thickness
32	Text Height	
34	Text Width	
36	Reserved	
38	Number of Contours (Characters)	
40	Pitch Extended	Height Extended
42	Cap Height	
44-47	Font Number ⋮	
48-63	Font Name ⋮	
64	Scale Factor	
66	X Resolution	
68	Y Resolution	
70	Master Underline Position	
72	Master Underline Thickness	
74	OR Threshold	
76	Global Italic Angle	
78-85	Character Complement	

Table 11-5 Format 11 Font Header (for Intellifont Unbound Scalable Fonts) (continued)

Desc. Size-2	Global Intellifont Data Size	
Desc. Size	Global Intellifont Data ⋮	
n	Copyright (optional) ⋮	
	Reserved (0)	Checksum

Table 11-6 Format 15 Font Header (for TrueType Scalable Fonts)

Byte	15 (MSB) 8	7 (LSB)0
0	Font Descriptor Size (minimum 72)	
2	Header Format (15)	Font Type
4	Style MSB	Reserved
6	Baseline Position	
8	Cell Width	
10	Cell Height	
12	Orientation	Spacing
14	Symbol Set	
16	Pitch (default HMI)	
18	Height	
20	x-Height	
22	Width Type	Style LSB
24	Stroke Weight	Typeface LSB
26	Typeface MSB	Serif Style
28	Quality	Placement
30	Underline Position (Distance)	Underline Thickness

Table 11-6 Format 15 Font Header (for TrueType Scalable Fonts) (continued)

32	Text Height	
34	Text Width	
36	First Code	
38	Last Code/Number of Characters	
40	Pitch Extended	Height Extended
42	Cap Height	
44-47	Font Number ⋮	
48-63	Font Name ⋮	
64	Scale Factor	
66	Master Underline Position	
68	Master Underline Thickness	
70	Font Scaling Technology	Variety
72	*[additional data may be inserted here]* ⋮	
Desc. Size	Segmented Font Data ⋮	
# - 2	Reserved (0)	Checksum

Data Types

In the font header and character descriptor information that follows, the abbreviations shown below are used to define the data type of each field:

Table 11-7 Font Header Field Data Type Notation

(B)	: Boolean	(0, 1)
(UB)	: Unsigned Byte	(0 . . 255)
(SB)	: Signed Byte	(-128 . . 127)
(UI)	: Unsigned Integer	(0 . . 65535)

Table 11-7 Font Header Field Data Type Notation (continued)

(SI)	: Signed Integer	(-32768 . . 32767)
(ULI)	: Unsigned Long Integer	(0 . . 2^{32}-1)
(SLI)	: Signed Long Integer	(-2^{31} . . 2^{31}-1)
(ASCxx)	: ASCII string	array (0 . . xx-1) of characters

Font Descriptor Size (UI)

Specifies the number of bytes in the font descriptor. See the font header figure for the appropriate font descriptor size.

Header Format (UB)

The Header Format byte identifies the font to format (see below).

Table 11-8 Header Format Values

Value	Format
0	PCL Bitmap
10	Intellifont Bound Scalable
11	Intellifont Unbound Scalable
15	TrueType Scalable (bound or unbound)
20	Resolution-Specified Bitmap

Font Type (UB)

Font type describes the font's relation to symbol sets.

Table 11-9 Font Type Values

Value	Font Type
0	Bound font. Character codes 32 to 127 [decimal] are printable.
1	Bound font. Character codes 32 to 127 [decimal] and 160 to 255 [decimal] are printable.

Table 11-9 Font Type Values (continued)

2	Bound font. All character codes 0 to 255 are printable, except 0, 7 to 15, and 27 [decimal] (see note below).
10	Unbound font. Character codes correspond to HP MSL numbers (for Intellifont unbound scalable fonts).
11	Unbound font. Character codes correspond to Unicode numbers (for TrueType unbound scalable fonts).

Note Access to those codes which are unprintable, yet have a character defined, requires the use of the Transparent Print Data command (refer to Chapter 8 for more information).

Style MSB (UI)

The Style MSB (byte 4) is combined with the Style LSB (byte 23) to make the style word. The contents of the style word are described below. The Style word (decimal) is calculated using the formula:

Style Word = Posture + (4 x Width) + (32 x Structure)

The binary structure of the Style word is shown below.

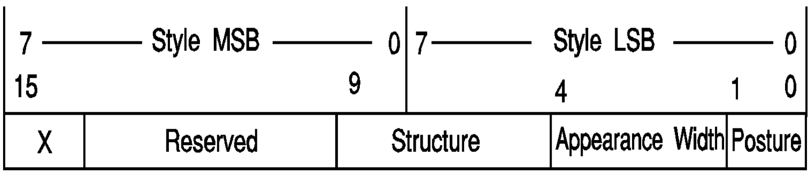


Table 11-10

Value	Posture (StyleWord partial sum)
0	Upright
1	Italic
2	Alternate Italic
3	Reserved

Table 11-11

Value	Appearance Width (multiply by 4 for StyleWord partial sum)
0	Normal
1	Condensed
2	Compressed or Extra Condensed
3	Extra Compressed
4	Ultra Compressed
5	Reserved
6	Extended or Expanded
7	Extra Extended or Extra Expanded

Table 11-12

Value	Structure (multiply by 32 for StyleWord partial sum)
0	Solid
1	Outline
2	Inline
3	Contour, Distressed (edge effects)
4	Solid with Shadow
5	Outline with Shadow
6	Inline with Shadow
7	Contour with Shadow
8-11	Patterned (complex patterns, subject to type family)
12-15	Patterned with Shadow
16	Inverse
17	Inverse in Open Border
18-30	Reserved
31	Unknown Structure

Note The reserved bits (15 - 10) should be set to zero.

If a value is requested, and a match not made, the request is ignored and the current font selection process continues as if the parameter was never requested (but it is saved in the attribute table).

Example

Assuming a font style of "italic compressed contour" is desired, the value (#) would be:

$$1 + (2\times4) + (3\times32) = 105$$

Baseline Position (UI)

Bitmap Font - Specifies the distance from the top of the cell to the baseline. The baseline is the dot row on which all of the characters in a given line appear to stand (see). The measurement of this distance is in font resolution dots, as defined in the Resolution Field of a Format 20 font header (default=300 dpi).

Intellifont Scalable - Specifies a Y-coordinate in the design window (refer to Figure 11-2.)

TrueType Scalable - Baseline Position must be set to zero.

Cell Width (UI)

Specifies the width of the cell. The cell must be wide enough to accept the widest character. The cell width range is 1 to 65535.

Bitmap Font - Specified in PCL coordinate system dots.

Scalable Font - Specified in design units.

Cell Height (UI)

Specifies the height of the cell. The design cell for a font must be tall enough to accept the tallest character and greatest descender. The legal range is 1 to 65535.

Bitmap Font - Specified in PCL coordinate system dots.

Scalable Font - Specified in design units.

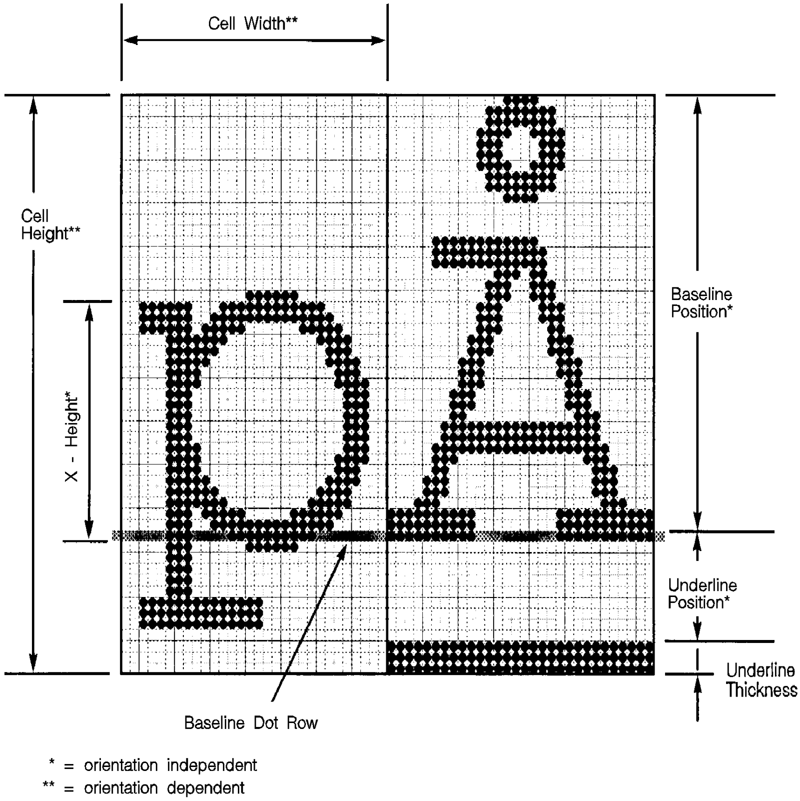


Figure 11-3 Character Cell - Bitmap

Orientation (UB)

Specifies the orientation of the font. All characters within the font must have the same orientation as those specified in the font header; otherwise they are discarded as they are downloaded.

0 = portrait (0 degrees; the orientation of the raster scan of the printer)
1 = landscape (90 degrees counterclockwise)
2 = reverse portrait (180 degrees counterclockwise)
3 = reverse landscape (270 degrees counterclockwise)

Bitmap Font - Unsupported values invalidate font creation.

Scalable Font - set to zero.

Note Hewlett-Packard recommends that bitmap soft fonts be designed in portrait (0°), using the paper motion and raster scan direction of the HP LaserJet Plus and LaserJet series II printers. The HP LaserJet IID, IIP, 2000, and PCL 5 LaserJet printers rotate the fonts to match the paper's physical coordinate system for the various paper sizes.

Spacing (B)

Specifies the spacing of the font. A value of zero (0) specifies fixed spacing and one (1) specifies proportional spacing.

Symbol Set (UI)

Specifies the symbol set for the font. This value is computed by taking the value of the value field for the symbol set, multiplying it by 32, adding the decimal (ASCII) value of the termination character (the symbol set ID character value) of the escape sequence, and subtracting 64.

$$\text{Font Descriptor Symbol Set Value} = \left(\begin{array}{l}\text{Escape Sequence}\\ \text{Value Field Value X 32}\end{array}\right) + \left(\begin{array}{l}\text{Decimal Value of Escape Sequence}\\ \text{Termination Character} - 64\end{array}\right)$$

For example, to compute the value for the ASCII (ISO-6) symbol set (Value = **0**, ID = U):

0U = (0×32) + (85–64) = 21

The legal range of symbol set escape sequence field values is 0 to 2047. Refer to Appendix C in the *PCL 5 Comparison Guide* for the HP defined symbol set values for use in the font header.

HP reserves the right to define the symbol set escape sequence value field values of 0 to 1023. Symbol set escape sequence value field values 1024 to 2047 are available for use by independent font vendors.

Symbol set escape sequence termination characters can be any upper case ASCII character “A” through “Z.” “Q” is reserved for use with HP “Specials” symbol sets and is not recommended for general use.

Notes

Symbol set ID’s of “@” and “X” do not have a corresponding Symbol Set selection command, sets marked as such can be selected only with the Font ID.

This field must have a value of 56 for a type 10 or 11 font (unbound Intellifont scalable) to be valid.

Pitch (UI)

Bitmap Font - Specifies the pitch of the font in quarter dots (four quarter-dot units, also known as radix dots, equal one dot). It combines with Pitch Extended to specify the pitch of the font in 1024^{th}-dots. Pitch defines the default HMI for the font.

For example, a a 17 cpi font designed at 300 dpi has a pitch value of 70 radix dots as calculated:

$$\frac{1\ \text{inch}}{17\ \text{char.}} \times \frac{300\ \text{dots}}{\text{inch}} \times \frac{4\ \text{radix-dots}}{\text{dot}} = 70.588\ \text{radix dot}$$

The remainder 0.588 is converted back to dots and then to 1024^{th}-dots as shown below:

$$\frac{0.588\ \text{radix dot}}{4\ \text{radix dot}} \times \frac{1024\ \text{units}}{\text{dot}} = 150\ \text{units/dot}$$

Pitch Extended is set to 150 1024^{ths}-units.

Note

For a proportional font, the width “printed” for a control code Space is determined by the pitch value, unless an HMI command is received following the selection of the font.

Scalable Fonts - Contains the master design space width (escapement) of the font in design units.

Height (UI)

Bitmap Font - Specifies the design height of the font in quarter-dots (radix dots). This value, converted to points, is used as the height characteristic value of the font. A PCL point is $\frac{1}{72}$ (0.01389) inch. It combines with Height Extended to specify the design height of the font in 1024th-dot (fonts designed at 300 dpi).

For example, a 10 point font at 300 dpi has a height of 166 quarter-dots (radix dots) (1200 quarter dots/inch, $\frac{1}{72}$ inch/point) as calculated:

$$\frac{10 \text{ point}}{} \times \frac{1 \text{ inch}}{72 \text{ point}} \times \frac{300 \text{ dots}}{\text{inch}} \times \frac{4 \text{ quarter-dots}}{\text{dot}} = 166.667$$

The remainder 0.667 is converted back to dots and then to

1024th-dot for a value of 170 1024th-dot for the Height Extended field (similar to that shown in the example for Pitch, above).

Intellifont Scalable - Specifies the master design height of the font in $\frac{1}{8}$ points. A typical value for this field is 2000.

TrueType Scalable - Set the Height field to zero.

xHeight (UI)

Bitmap Font - Specifies the height of the lower case “x” in quarter-dots (radix dots).

Scalable Fonts - Specifies the distance from the baseline to the lower case “x” height in design units.

Width Type (SB)

Specifies the proportionate width of characters in the font.

Table 11-13 Width Type Values

Value	Width Type
-5	Ultra Compressed
-4	Extra Compressed
-3	Compressed or Extra Condensed
-2	Condensed
0	Normal
2	Expanded
3	Extra Expanded
Additional width types may be added by HP.	

Style LSB (UB)

The least significant byte (LSB) of the Style word. Refer to Style MSB for a description of the Style word.

Stroke Weight (SB)

Specifies the thickness of the strokes used in designing the font. The supported stroke weight values are -7 through 7. The thinnest stroke available is -7; the thickest stroke weight is 7. The standard stroke weight for a medium font is 0; the standard stroke weight for a bold font is 3; and, the standard stroke weight for a light font is -3.

Table 11-14 Stroke Weight Values

Value	Stroke Weight
-7	Ultra Thin
-6	Extra Thin
-5	Thin
-4	Extra Light

Table 11-14 Stroke Weight Values (continued)

-3	**Light**
-2	Demi Light
-1	Semi Light
0	**Medium, Book, or Text**
1	Semi Bold
2	Demi Bold
3	**Bold**
4	Extra Bold
5	Black
6	Extra Black
7	Ultra Black

Typeface (UB)

This field specifies the HP typeface number of the font. The current version of this field, supported by the &payette; printer, is described first. Then a previous field, supported in earlier printers, is described.

Current Usage

In the LaserJet 4 printer version of this field, an unsigned short integer is assembled from the two unsigned bytes of data. Printers, when seeking to match a typeface request with available font resources, may treat the typeface number as a single value. If an exact match cannot be made, the request may be ignored (for selection purposes, however, the font select table is updated).

The procedure for allocating typeface numbers for the font products of various vendors, however, will consider the typeface number to be composed of two distinct fields: a vendor field (consisting of the four most significant bits) and a typeface family field (consisting of the 12 least significant bits). The following diagram illustrates this scheme:

Table 11-15 Typeface Family Value (Current)

15 12	11 0
Vendor	Typeface Family

Vendor Number - Bits 15 - 12. This value is assigned by HP and is between decimal values 0 and 15.

Table 11-16 Current Vendor Number Values

Value	Vendor
= 0	Reserved
= 1	Agfa Division, Miles Inc.
= 2	Bitstream Inc.
= 3	Linotype Company
= 4	The Monotype Corporation plc
= 5	Adobe Systems Inc.
= 6-15	(Reserved)

Typeface Family Number - Bits 11 - 0 This value is between 0 and 4095. See Appendix C in the *PCL 5 Comparison Guide*.

Typeface Family Values are calculated according to the following formula:

$$\text{Typeface Base Value} + \left(\text{Vendor Value} \times 4096\right) = \text{Typeface Family}$$

Example

The HP typeface number for Agfa Dom Casual typeface is 4157 (vendor value=1, and typeface value=61):

$$\mathbf{61} + (\mathbf{1} \times 4096) = 4157$$

Previous Usage

The previous treatment of the Typeface field supported the LaserJet IIP, IID and LaserJet III family printers. It consisted of the Typeface Least Significant Byte (LSB; the original, one-byte typeface value used prior to the LaserJet IID printer) and the Typeface Most Significant Byte (MSB) in the font header.

The previous typeface family value field is shown below. It included a 4-bit field to specify the vendor number, a 2-bit field for the version number, and a 9-bit field which contained the typeface base number. The most significant bit of the MSB was always zero.

Table 11-17 Typeface Family Value (Previous)

15	14	10	8 0
0	Vendor	Version	Typeface Base Value

Table 11-18 Previous Vendor Number Values

Value	Vendor
0,1	Reserved
2	Agfa Division, Miles Inc.
4	Bitstream Inc.
6	Linotype Company
8	The Monotype Corporation plc
10	Adobe Systems Inc.
3,5,7,9,11-15	(Reserved)

Vendor-Version The Vendor-version (bits 10 and 9) value was from 0 to 3. It changed when the vendor changed the width or design of the characters in a font.

Typeface Base Value The Typeface Base Number (bits 0 through 8) ranged from 0 to 511. Some of these values referred to the styles that vary by structure and appearance width (such as Helvetica Condensed, Helvetica Outline, etc.). Do not use these values in new designs since they are being deleted. Refer to Appendix C in the *PCL 5 Comparison Guide* for a list of typeface families and their typeface base values.

Note For future compatibility, use the two-byte (typeface MSB/LSB) typeface family value. All scalable fonts use the larger typeface family value. Older bitmap fonts use the smaller typeface base value.

Serif Style (UB)

Specifies one of the following defined serif styles.

Serif Style values 0-63 (the lower six bits of the style field) are ignored by the printer for bitmap fonts. However, the upper two bits (bits 6 and 7) are used by a scalable font header to determine the serif style of the typeface insensitive characters to complement the font. Serif style values for the lower six bits are listed in the table below. Serif style values for the upper two bits are listed in the following table.

Table 11-19 Serif Style Values

Value	Serif Style
0	Sans Serif Square
1	Sans Serif Round
2	Serif Line
3	Serif Triangle
4	Serif Swath
5	Serif Block
6	Serif Bracket
7	Rounded Bracket
8	Flair Serif, Modified Sans
9	Script Nonconnecting
10	Script Joining

Table 11-19 Serif Style Values (continued)

11	Script Calligraphic
12	Script Broken Letter
13-63	Reserved
Values for bits 6 & 7	
64	Sans Serif
128	Serif
192	Reserved

Quality (UB)

This field specifies the quality of the font.

Table 11-20 Quality Values

Value	Quality
0	Data processing (draft)
1	Near Letter Quality
2	Letter Quality

Placement (SB)

Placement specifies the position of character patterns relative to the baseline.

Bitmap Font - The placement values for bitmap fonts are listed in the following table.

Table 11-21 Bitmap Font Placement Values

Value	Placement
1	Superior
0	Normal
-1	Inferior

Scalable Font - Set the Placement field to zero.

Underline Position (Distance) (SB)

Bitmap Font - Specifies the distance from the baseline to the top dot row of the underline in font design dots. Zero specifies an underline position at the baseline. A positive value specifies an underline position above the baseline. A negative value specifies an underline position below the baseline.

Scalable Font - Set Underline Position to zero. The Master Underline Position field (see below) identifies this information for scalable fonts.

Underline Thickness (UB)

Specifies the thickness of the underline in font design dots for a bitmap font.

Bitmap Font - A bitmap font prints three-dot thick underlines at 300 dpi (six-dot thick at 600 dpi).

Scalable Font - Set Underline Thickness to zero. The Master Underline Thickness field (see below) identifies this information for scalable fonts.

Text Height (UI)

Specifies the font's optimum inter-line spacing. This value is typically equal to 120% of the height of the font.

Bitmap Font - Specified in quarter-dots (radix dots).

Scalable Fonts - Specified in design units.

Text Width (UI)

Specifies the font's average lowercase character width. (This average width may be weighted on the basis of relative frequency.)

Bitmap Font - Specified in quarter-dots (radix dots).

Scalable Font - Specified in design units.

First Code (UI)

First Code specifies the character code of the first printable character in the font. This value is between 0 and 255 inclusive. The Space Character may be printable and will print an image if one is defined, otherwise a Space control code is executed. Currently, PCL 5 LaserJet printers use the Font Type field to determine the first and last codes of the symbol set, as shown below:

Table 11-22

Font Type	First Code../..Last Code
0	32/127
1	32/127 - 160/255
2	0/255
10	Set to 0 (for unbound font)
11	Set to 0 (for unbound font)

Last Code / Number of Characters (UI)

Bound Font: Specifies the last code in the font. This value may be greater than the last code of the symbol set as implied by the font type because there may be components of compound characters that are not part of the symbol set but must be downloaded. The printable codes are implied by the font type (refer to first code described above).

Unbound Font: For an unbound font (type 10 or 11), this field specifies the maximum number of characters that can be downloaded into the font.

Pitch Extended (UB)

Bitmap Font - This is an addition to the Pitch field which extends the pitch an extra eight bits. The value of this field is in font design units. For example, a 17 pitch font designed at 300 dpi has a Pitch field of 70 (17.5 dots, or 17.1429 cpi) and a Pitch Extended field of 150 (0.1465 dots additional, which adds to 17.6465 dots, or 17.0005 pitch).

An example for calculating the Pitch and Pitch Extended fields is provided in the Pitch field description, above.

Scalable Font - Set Pitch Extended field to zero.

Height Extended (UB)

Bitmap Font - This is an addition to the Height field which extends the height an extra eight bits. The value of this field is in font design units. For example, a 10 point font designed at 300 dpi would have a Height field of 166 (41.5 dots, or 9.96 points) and a Height Extended field of 170 (0.1660 dots additional, which adds to 9.9998 points). This field is calculated similar to the Pitch Extended field. Refer to the Pitch description, above.

Scalable Font - Set The Height Extended field to zero.

Cap Height (UI)

Cap Height is a percentage of the Em of the font and is used to calculate the distance from the capline (top of an unaccented, uppercase letter, such as an "H") to the baseline.

Bitmap Font - Fonts containing a 0 in this field are assumed to have a cap height percentage of 70.87% of Em (Em being a measure, in points, of the height of the body of the font.

The Cap Height data is represented as the product of the cap height percentage and the maximum unsigned integer:

$$0.7087 \times 65535 = 46{,}445$$

For non-zero values, the Cap Height % is calculated as follows:

$$\% = \frac{\text{Cap Height Data}}{65535} \text{ X } 100$$

Scalable Font - Contains the cap height in design units.

Font Number (ULI)

The Font Number field uses four bytes (byte 44, 45, 46, and 47). The lower three bytes (45, 46, and 47) contain the font number in hexadecimal. This is the number the vendor assigns to their typeface. The most significant byte (byte 44) consists of a flag in the most significant bit indicating whether the font is in its native (0) format or has been converted (1) from another format. The remaining lower seven bits contain the ASCII decimal value for the first initial of the font vendor's name (this is assigned by Hewlett-Packard). The following initials have been assigned:

Table 11-23

Initial	HexValue	Vendor Name
A	41	Adobe Systems Inc.
B	42	Bitstream Inc.
C	43	&AGFA;
H	48	Bigelow && Holmes
L	4C	Linotype Company
M	4D	Monotype Corporation plc

For example, the number that Agfa assigns for a CG Times Bold Italic, native format, font is 92505. This number is converted to hexadecimal and used for the lower three bytes of the Font Number. Bit 8 of byte 44 is 0, since the native format is used and the lower seven bits are the ASCII value for "C" (C for Compugraphic; 0100 0011). This process is summarized below.

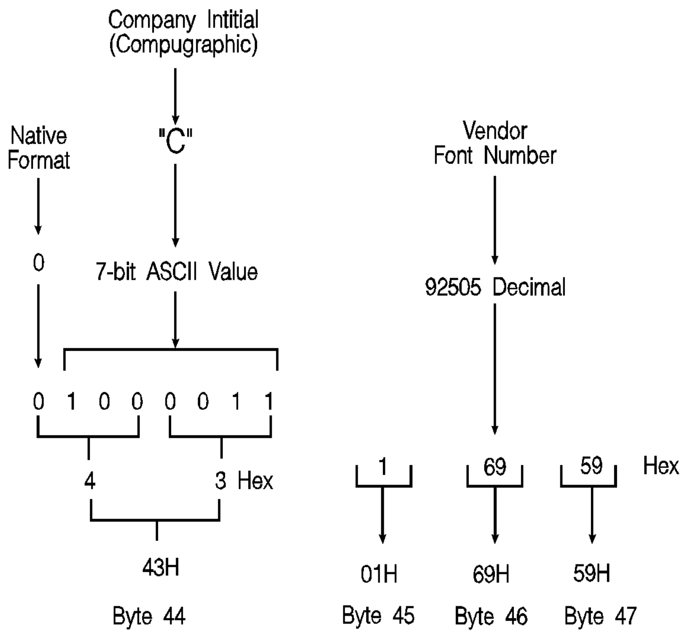


Figure 11-4

This field is ignored by the printer for bitmap fonts.

Font Name (ASC16)

This is a 16 character ASCII field to which you may assign a font name. The font name is used in the Typeface List (or Font List printout), under **Name** or **Typeface** (if the printer does not have a name string assigned to the typeface family code in its font selection table).

X Resolution (UI)

The X Resolution field is the pixel resolution in the X scan direction at which the font was designed.

Note This field is not present in the Format 15 Font Header and is not necessary for TrueType fonts.

Y Resolution (UI)

The Y Resolution field is the pixel resolution in the Y scan direction at which the font was designed.

Note This field is not present in the Format 15 Font Header and is not necessary for TrueType fonts.

Scale Factor (UI)

The Scale Factor field indicates the number of design units per Em, and is the unit used for all scalable metrics in the font header. It is used for TrueType and Intellifont scalable soft fonts.

Master Underline Position (SI)

The Master Underline Position is the top of the PCL floating underline with respect to the baseline in design units.

Note For scalable fonts, the Master Underline Position field replaces the 1-byte Underline Position field.

Master Underline Thickness (Height) (UI)

The Master Underline Thickness field defines thickness of the floating underline in design units.

Font Scaling Technology (UB)

For scalable fonts, this field species the technology to be used for font scaling.

Table 11-24

Value	Font Scaling Technology
0	Intellifont
1	TrueType

Variety (UB)

The interpretation of this field depends on the value of the preceding (Font Scaling Technology) byte. For TrueType fonts, this field must be set to zero.

OR Threshold (UI)

Formerly called the “LRE Threshold,” this is the pixel size in design units above which the missing pixel recovery process is switched on in Intellifont scaling and rasterization.

Note The size of a pixel (in design units) increases as point size and device resolution decrease.

Global Italic Angle (SI)

The Global Italic Angle field contains the tangent of the italic angle times 2^{15} (relative to the vertical). Set this field to zero for upright fonts. For detailed information on Global Italic Angle, refer to *Intellifont Scalable Typeface Format*.

Global Intellifont Data Size (UI)

The Global Intellifont Data Size identifies the size of the Global Intellifont data block. For detailed information about Global Intellifont Data Size, refer to *Intellifont Scalable Typeface Format*.

Global Intellifont Data

For detailed information on Global Intellifont Data refer to *Intellifont Scalable Typeface Format*.

Character Complement (Array of UB)

This 8-byte field qualifies the compatibility of a type 10 or 11 font with various character sets. Each bit is independently interpreted with the exception of the least significant three bits. (Bit 63 refers to the most significant bit of the first byte, and bit 0 refers to the least significant bit of the eighth byte.)

Note In the Format 15 Font Header for TrueType Scalable Fonts, the data in this field is contained in the "CC" (Character Complement) field in the Segmented Font Data section immediately following the descriptor data. See "Segmented Font Data" later in this chapter

.

Table 11-25 MSL Symbol Index

Bit Field	Designated Use
58-63	Reserved for Latin fonts.
55-57	Reserved for Cyrillic fonts.
52-54	Reserved for Arabic fonts.
50-51	Reserved for Greek fonts.
48-49	Reserved for Hebrew fonts.
3-47	Miscellaneous uses (South Asian, Armenian, other alphabets, bar codes, OCR, Math, PC Semi-graphics, etc.).
0-2	Symbol Index field. 111 - MSL Symbol Index

Table 11-26 Unicode Symbol Index

Bit Field	Designated Use
32-63	Miscellaneous uses (South Asian, Armenian, other alphabets, bar codes, OCR, Math, etc.).
28-31	Reserved for Latin fonts.
22-27	Reserved for platform/application variant fonts.
3-21	Reserved for Cyrillic, Arabic, Greek and Hebrew fonts.
0-2	Symbol Index field. 110 - Unicode Symbol Index

Individually defined bits are shown in the following two tables:

Table 11-27 MSL Symbol Index Character Complement Bits

Bit	Value
63	0 if font is compatible with standard Latin character sets (e.g., Roman-8, ISO 8859-1 Latin 1); 1 otherwise.
62	0 if font is compatible with East European Latin character sets (e.g., ISO 8859-2 Latin 2); 1 otherwise.
61	0 if font contains Turkish character sets (e.g., ISO 8859/9 Latin 5); 1 otherwise.
34	0 if font has access to the math characters of the Math-8, PS Math and Ventura Math character sets; 1 otherwise.
33	0 if font has access to the semi-graphic characters of the PC-8, PC-850, etc. character sets; 1 otherwise.
32	0 if font is compatible with ITC Zapf Dingbats series 100, 200, etc.; 1 otherwise.
2, 1, 0	111 if font is arranged in MSL Symbol Index order.

Table 11-28 Unicode Symbol Index Character Complement Bits

Bit	Value
31	0 if font is compatible with 7-bit ASCII; 1 otherwise.
30	0 if font is compatible with ISO 8859/1 Latin 1 (West Europe) character sets; 1 otherwise.
29	0 if font is compatible with ISO 8859/2 Latin 2 (East Europe) character sets; 1 otherwise.
28	0 if font is compatible with Latin 5 (Turkish) character sets (e.g., ISO 8859/9 Latin 5, PC-Turkish); 1 otherwise.
27	0 if font is compatible with Desktop Publishing character sets (e.g., Windows 3.1 Latin 1, DeskTop, MC Text); 1 otherwise.

Table 11-28 Unicode Symbol Index Character Complement Bits

26	0 if font is compatible with character sets requiring a wider selection of accents (e.g., MC Text, ISO 8859/1 Latin 1); 1 otherwise.
25	0 if font is compatible with traditional PCL character sets (e.g., Roman-8, Legal, ISO 4 United Kingdom); 1 otherwise.
24	0 if font is compatible with the Macintosh character set (MC Text); 1 otherwise.
23	0 if font is compatible with PostScript Standard Encoding (PS Text); 1 otherwise.
22	0 if font is compatible with Code Pages (e.g., PC-8, PC 850, PC-Turk, etc.); 1 otherwise.
2,1,0	110 if font is arranged in Unicode Symbol Index order.

There are no invalid Character Complement field values. Examples of values for the field include:

Table 11-29

Bit Field	Designated Use
Value (hex)	**Meaning**
MSL:	
“`0000000000000000`”	Default complement; font is compatible with any character set.
“`7fffffffffffffff`”	Font is indexed in MSL and is compatible only with standard West Latin character sets.
“`ffffffffeffffffff`”	Font is indexed in MSL and is compatible only with ITC Zapf Dingbat character sets.
Unicode:	
“`fffffffff3fffffe`”	Font is indexed in Unicode and is compatible only with standard West Latin character sets.

Table 11-29 (continued)

"`ffffffff5ffffffe`"	Font indexed in Unicode and is compatible only with East Europe Latin character sets.

Checksum

The Checksum field is over bytes 64 through the end of the header. The checksum should contain a value which, when added to the sum of byte 64 through the reserved byte, equals a value which, when divided by 256 (modulo 256 arithmetic), results in a remainder of 0. For example, if the sum = 10,234 then, 10,234 mod 256 = 250. Therefore, the checksum should = 6 (since 250+6 = 256 which would produce 0 [mod 256]).

Note In the Format 15 Font Header for TrueType Scalable Fonts, this field is located at the end of the Segmented Font Data section immediately following the descriptor data. See "Segmented Font Data" later in this chapter.

Copyright

This field contains ASCII data and is optional.

Note In the Format 15 Font Header for TrueType Scalable Fonts, this field is located in the Segmented Font Data section immediately following the descriptor data. See "Segmented Font Data" later in this chapter.

Segmented Font Data (Format 15)

The Segmented Font Data section immediately follows the main body of a Format 15 Header for TrueType Scalable Fonts. Each segment contains three parts: a **Segment Identifier, Segment Size,** and **Data Segment.**

The Segmented Font Data section is terminated by the Null Segment. (In the deviant case where no Null Segment is encountered prior to the end of the font header &emdash; as defined in the Font Header command &emdash; the font is invalidated. A font also is invalidated in the event that a Null Segment is encountered too soon.)

Table 11-30 below shows the structure of the Segmented Font Data section.

Table 11-30 Segmented Font Data

Byte	15 (MSB) 8	7 (LSB) 0
x + 0	First segment, Segment Identifier	
x + 2	First segment, Segment Size	
x + 4 ⋮	First segment, Data Segment ⋮	
x + 4 + 1st seg size	Second segment: Segment Identifier, Size, Data Segment ⋮	
⋮	⋮	
# – 6	Null Segment Identifier (FFFF - hex)	
# – 4	Null Segment Size (0)	
# – 2	Reserved	Checksum
x = Font Descriptor Size. # = Font header length (as defined in Font Header command).		

Segment Identifier (UI)

Each entry in the Segmented Font Data Section has its own unique identification number. The following values are defined:

Table 11-31

Value	Mnemonic	Data Segment
17219	CC	Character Complement
17232	CP	Copyright
18260	GT	Global TrueType Data
18758	IF	Intellifont Face Data
20545	PA	PANOSE Description
22618	XW	XWindows Font Name
65535		Null Segment

Data segments with an unrecognized identifier are ignored.

Segment Size (UI)

For each entry in the Segmented Font Data section, the Segment Size indicates the number of bytes in the immediately following Data Segment. The size for the Null Segment is 0.

Formats of Data Segments

AP (Application Support Segment) The definition of this segment is reserved.

CC (Character Complement) This field has the same form (*i.e.,* 8 unsigned bytes) and function as does the Character Complement of Format 11 fonts. The Character Complement field should be present with type 10 and 11 (unbound) fonts, but has no role to play in type 0, 1 and 2 (bound) fonts.

CP, copyright This field will consist of ASCII data and is optional.

GI (Global Intellifont Data) Reserved for future use.

GT (Global TrueType Data) This data segment contains first a Table Directory, then five or more tables used by the TrueType font scaler. Every TrueType font needs to have this segment.

The Table Directory is patterned after the initial segment of the TrueType font file as described in *True Type Font Files*. The Table Directory has a 12-byte header and 16 bytes per entry in the Table Directory. The Table Directory is organized in alphabetical order by the 4-byte table names. For each entry, there is an offset relative to the beginning of the soft font's Global TrueType Data Segment.

The **Global TrueType Data** for every TrueType font entity must contain a **head, hhea, hmtx** and **maxp** table.

Another required table is the **gdir** table. When the font header is downloaded, the **gdir** table should have a size of 0 and an offset of 0. The **gdir** table is then built in RAM to accommodate the maximum number of glyphs to be downloaded to the given font — with 2 or 4 bytes of offset and 2 bytes of length per glyph. This maximum number of glyphs is obtained from the numGlyphs field of the **maxp** table. Entries in the **gdir** table are filled in by the TrueType rasterizer as characters are downloaded.

The optional **cvt, fpgm** and **prep** tables, as defined in *True Type Font Files*, typically appear in the Global TrueType Data Segments of hinted TrueType soft fonts, but should not appear in unhinted fonts.

IF (Intellifont Face Data) Reserved for future use.

PA (PANOSE Description) This data segment of variable length may be used for the purpose of font selection and substitution. Its definition continues to evolve. A 10-field (10-byte) version sufficient for the description of most Latin fonts appears under the OS/2 table in *True Type Font Files*.

PF (PS-Compatible Font Name) Reserved for future use.

XW (x-windows font name) This ASCII field contains standard X-Windows font names.

Checksum

The value of this byte, when added to the sum of all of the bytes from byte 64 of the descriptor through the Reserved byte, should equal 0 in modulo 256 arithmetic.

Font Header Examples

Two examples for downloading a Font Header are provided below; one for a bitmap font and one for an Intellifont scalable font.

Bitmap Example

To download a bitmap font header for a portrait HP Roman-8, 10 pitch, 12 point, upright, medium, Courier font, with an ID number of one, send:

$^{E}{}_{C}$***c1D** (set Font ID to 1)

$^{E}{}_{C}$**)s#W** (# = 64 bytes of font descriptor data + *x* bytes of optional data)

An example of the bitmap header is shown on the following page.

Table 11-32

FIELD NAME	VALUE	DESCRIPTION
Font Descriptor Size	64	Bytes
Header Format	0	Bitmap Font Format
Font Type	1	Eight Bit
Style MSB	0	
Reserved	0	
Baseline Position	40	
Cell Width	30	
Cell Height	53	
Orientation	0	Portrait
Spacing	0	Fixed Pitch
Symbol Set	277	8U: Roman-8
Pitch	120	Quarter Dots (30.00 Dots)
Height	200	Quarter Dots (50.00 Dots)
x Height	88	Quarter Dots (22.00 Dots)
Appearance Width	0	Normal

Table 11-32 (continued)

Style LSB	0	Upright, Normal Width, Solid (0,0,0)
Stroke Weight	0	Medium
Typeface LSB	3	Body Text
Typeface MSB	0	No Font Vendor ID
Serif Style	2	Serif Line
Quality	0	∅
Placement	0	∅
Underline Position	-10	∅
Underline Thickness	3	∅
Text Height	200	Quarter dots (50.00 Dots)
Text Width	120	Quarter Dots (30.00 Dots)
First Code	33	∅
Last Code	254	∅
Pitch Extended	0	∅
Height Extended	0	∅
Cap Height	36713	56.02% of Em
Font Number	0	No Font Vendor Number
Font Name	Courier	
....Copyright Statement .... (optional) }= *x* bytes added to header data		
....Application Support ....(optional) }= *x* bytes added to header data		

Intellifont Scalable Example

To download an Intellifont scalable header for an HP Roman-8, upright, medium, CG Times scalable font, with an ID number of one, send:

$^{E}_{C}$*c1D (set Font ID to 1)

$^{E}_{C}$)s#W (# = 80 bytes of font descriptor data + *x* bytes of Global Intellifont data + *x* bytes of optional data)

Table 11-33

FIELD NAME	VALUE	DESCRIPTION
Descriptor Size	80	Bytes
Header Format	10	Scalable Font Format
Font Type	1	Eight Bit
Style MSB	0	
Reserved	0	
Baseline Location	5380	Y reference in Design Window
Cell Width	0	..not defined for Intellifont
Cell Height	0	..not defined for Intellifont
Orientation	0	..not defined for scalable fonts
Spacing	1	Proportional
Symbol Set	277	8U: Roman-8
Pitch	2602	29.63% Em Default HMI
Design Height	2000	250 Points * 8
x-Height	4009	45.65% Em, 68.52% Cap Height
Appearance Width	0	Normal
Style LSB	0	Upright, Normal Width, Solid (0,0,0)
Stroke Weight	0	Medium or Text Weight
Typeface LSB	5	Times Roman (generic design family)
Typeface MSB	16	Agfa
Serif Style	134	Serif, Bracketed (2,6)
Quality	0	..not defined for scalable fonts

Table 11-33 (continued)

Placement	0	..not defined for scalable fonts
Underline Position	0	..not defined for scalable fonts
Underline Thickness	0	..not defined for scalable fonts
Text Height	0	..not defined for Intellifont
Text Width	4391	Width of En Space
First code	33	
Last Code	273	Compound Pieces Present (n255)
Pitch Extended	0	..not defined for scalable fonts
Height Extended	0	..not defined for scalable fonts
Cap Height	5851	66.7% Em
Font Number	hex 43 01 69 54	Native, Agfa, CG Times (0,C,92500)
Font Name	"CG Times "	(16 character ACSII field)
Scale Factor	8782	
X Resolution	2540	
Y Resolution	2540	
Master Underline Position	-1747	
Master Underline Thickness	449	
OR Threshold	176	
Global Italic Angle	0	
Global Intellifont Data Size	112	

Table 11-33 (continued)

....Global Intellifont Data		}= 112 bytes added to header data
....Copyright Statement (optional)		}= *x* bytes added to header data
....Application Support (optional)		}= *x* bytes added to header data

Character Definitions

Following the font header, the individual characters must be defined. Every PCL character definition contains a character descriptor and a body of character data. The character definition always consists of one or more character data blocks. Each character data block begins with its own header. The character data block header always has a size of 2 bytes.

The first data block of a character definition must always have a character descriptor immediately after its 2-byte header.

Character descriptor/data is downloaded using the Character Definition command preceding every character (see *Character Definition Command*).

Notes

A unique character code, using the Character Code command, must be designated prior to the download of a character descriptor and data. If the font being downloaded already contains a character with this code, the existing character is deleted during the download of the character descriptor and data.

Unless otherwise specified, inappropriate values in a character descriptor field invalidates the character download process; a character is not created, and the associated descriptor and data are discarded.

An undefined printable character is one which is in the printable range of the font type but has no defined pattern. Attempts to print an undefined printable character from a font result in the execution of a Space control code.

Character descriptor fields identified as “reserved” should be set to zero.

If the total byte count of the character descriptor and data exceeds 32767 bytes, then the remaining data must be sent using the continuation descriptor.

Character Code Command

The Character Code command establishes the decimal code that is associated with the next character downloaded. This value is used to reference the character for printing.

$^{E}{}_{C}$ *** c # E**

\# =character code

Default **=** 0
Range **=** 0 - 65535

Notes

For unbound fonts, the character code for a given character equals its symbol index value.

For TrueType fonts, a special code must be used to download glyphs which never stand alone as characters. FFFF (hex) should be used for this purpose.

Example

To designate the character code for an ASCII lower-case “p”, send:

$^{E}{}_{C}$***c112E**

Character Definition Command

The Character Descriptor and Data command is used to download character data blocks to the printer for both bitmap and scalable fonts.

$^{E}{}_{C}$ **(s # W** [character descriptor and data]

Default **=** N/A
Range **=** 0 - 32767

The value field (#) identifies the number of bytes in the immediately following character data block. The maximum number is 32767.

For a detailed description of the Character Descriptor fields for bitmap fonts refer to *Character Descriptor and Data Format for PCL Bitmap Fonts.* For Intellifont scalables, refer to "Character Descriptor and Data Format for Intellifont Scalable Fonts." For TrueType fonts, refer to "Character Descriptor and Data Format for TrueType Fonts."

Note Examples for defining a bitmapped portrait and landscape character are provided under *Character Definition Examples*, after the *Character Descriptor Formats* section, later in this chapter.

Character Descriptor Formats

Character definition formats for PCL Bitmap, Intellifont Scalable and TrueType Scalable fonts are shown on the following pages.

Note The following notation is used to define the data type of each field in the character descriptors.

Table 11-34 Character Descriptors/Data Continuation Block

(B)	: Boolean	(0,1)
(UB)	: Unsigned Byte	(0 . . 255)
(SB)	: Signed Byte	(-128 . . 127)
(UI)	: Unsigned Integer	(0 . . 65535)
(SI)	: Signed Integer	(-32768 . . 32767)

Character Descriptor and Data Format for PCL Bitmap Fonts

The descriptor of a PCL bitmap character is at least 14 bytes long and contains information such as the character's width and height.

The character data is binary (raster) data that identifies the shape of the character.

Table 11-35 shows the format of the bitmap character descriptor and data.

Table 11-35 PCL Bitmap Character Descriptor and Data Format

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (4)	Continuation (0)
2	Descriptor Size (14)	Class (1)
4	Orientation	Reserved (0)

Table 11-35 PCL Bitmap Character Descriptor and Data Format (continued)

6	Left Offset	
8	Top Offset	
10	Character Width	
12	Character Height	
14	Delta X	
16	Raster Character Data: (in bytes) ⋮	

Table 11-36 PCL Bitmap Continuation Character Descriptor and Data Format

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (4)	Continuation (non-zero)
2	Raster Character Data: (in bytes) ⋮	

Format (UB)

This is the first byte of every character data block header. It specifies the format of the character descriptor and data. The format number used for bitmap fonts is 4. This format must match that of the Font Header.

Table 11-37

Value	Format
4	LaserJet Family (Raster)
10*	Intellifont Scalable
15*	TrueType Scalable
* These are described later in this chapter.	

If the format number is different from that expected by the device, the character is discarded.

Continuation (B)

This is the second (and last) byte of every character data block header. It specifies whether the following data is the first (0) data block of a new character definition, or a continuation (1) block for a character definition which has already been received by the printer. Because the value field in a Character Definition command is limited to 32767 bytes, characters whose byte count exceed this must be sent in two or more blocks.

Descriptor Size (UB)

This is the first byte of the character descriptor. It specifies the size of the character descriptor in bytes. The descriptor size used by the HP LaserJet printer family for bitmap fonts is 14.

Class (UB)

Specifies the format of the character data. For bitmap fonts only values 1 and 2 are used, as described below.

Table 11-38

Value	Class
1	Bitmap
2	Compressed Bitmap
3*	Contour (Intellifont Scalable)
4*	Compound Contour (Intellifont Scalable)
15*	TrueType Scalable
* These are described later in this chapter.	

Class 1 - Bitmap Data

Class 1 or bitmap (raster) character data is a string of bytes containing the dot-per-bit image of the character, no data compression. If a bit is set to one, the corresponding dot is printed. The data is grouped in dot rows. A row describes a one-dot-high strip of the character from left to right, in the direction of the printer's raster scan (see the Portrait Bitmap Character Data Example, at the end of this chapter). Zeroed bits must be added to the end of each row to make it contain an integral number of bytes. The dot rows are organized from top to bottom of the character. For example, the first dot row of data corresponds to the top dot row of the character.

The number of bytes of the character data should be exactly Character Width (in bytes) times Character Height. If more

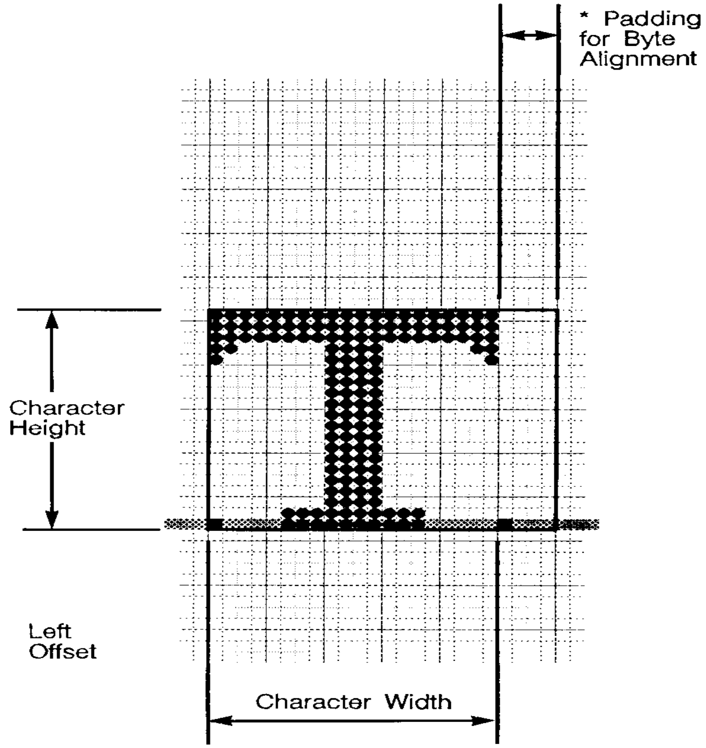
data is received, it is discarded; if less data is received, the character consists of only the data downloaded.

Class 2 - Compressed Bitmap Data

For a compressed bitmap character, the data is composed of a string of bytes using a run-length encoding with line repetition compressed format (see Figure 11-2). The first byte indicates the number of times the first raster row is repeated after its initial occurrence. It is assumed that the first pixel in a row is white, hence the second byte indicates how many white pixels start the row. The third byte indicates how many black pixels, the fourth byte indicates the number of white pixels again, etc. If the first pixel in a row is black, the white pixel indicator (the second byte) is 0. If there are more than 255 pixels in a row of the same type, there is a byte containing 255, followed by a 0 byte, followed by a byte containing the count of remaining pixels of the current type.

The width of each row is determined by the character width (in dots) as specified in the character descriptor for the character. The pixel count (number of 1's and 0's bits) for each row in the character cell must equal the character width. For example, in Figure 11-5, the cell width is 20, thus each row (excluding the repetition count byte) adds up to 20.

Once the row has been filled, the row is duplicated as indicated in its first byte, then a new row is started.



Line Repetition	# white pixels	# black pixels	# white pixels	# black pixels	# white pixels	# black pixels
2	0	20	-	-	-	-
0	0	2	6	4	6	2
0	0	1	7	4	7	1
12	8	4	8	-	-	-
1	5	10	5	-	-	-

Uncompressed - 60 Bytes
Compressed - 25 Bytes

* Byte alignment is necessary only for raster data i.e. (Not necessary for compressed raster data.)

Figure 11-5 Class 2 Character Data

Orientation (UB)

Orientation byte specifies the orientation of the character. The orientation of the character must match the orientation of the font.

Table 11-39

Value	Orientation
0	Portrait
1	Landscape
2	Reverse portrait
3	Reverse landscape

If the orientation is not supported or is different from the orientation specified in the font header, the character is discarded.

Left Offset (SI)

Left offset specifies the distance in dots from the reference point to the left side of the character pattern on the physical page coordinate system (this value is orientation dependent). The left and top offsets locate the character reference point about the cursor position (see Figure 11-6 and Figure 11-7).

PCL 5 printers support kerning (both negative left and right side bearings) of both fixed-pitch and proportionally-spaced fonts. Note that large offsets could place the character off the printable area of the page causing the character to be clipped.

The legal range for the left offset is –16384 to 16384 dots.

Top Offset (SI)

Top offset specifies the distance in dots from the reference point to the top of the character pattern on the physical coordinate system (this value is orientation dependent.) The left and top offsets locate the character reference point about the cursor position (see Figure 11-6 and Figure 11-7). The legal range for the top offset is –16384 to 16384 dots.

Character Width (UI)

The Character Width, used for bitmap fonts only, identifies the width of the character in dots on the physical coordinate system. Generally, this width is from the farthest left black dot to the farthest right black dot. Character width is orientation dependent.

The legal range for character width is 1 to 16384 dots.

Character Height (UI)

Character Height specifies the height of the character in dots on the physical coordinate system. Character height is orientation dependent.

The legal range for character height is 1 to 16384 dots.

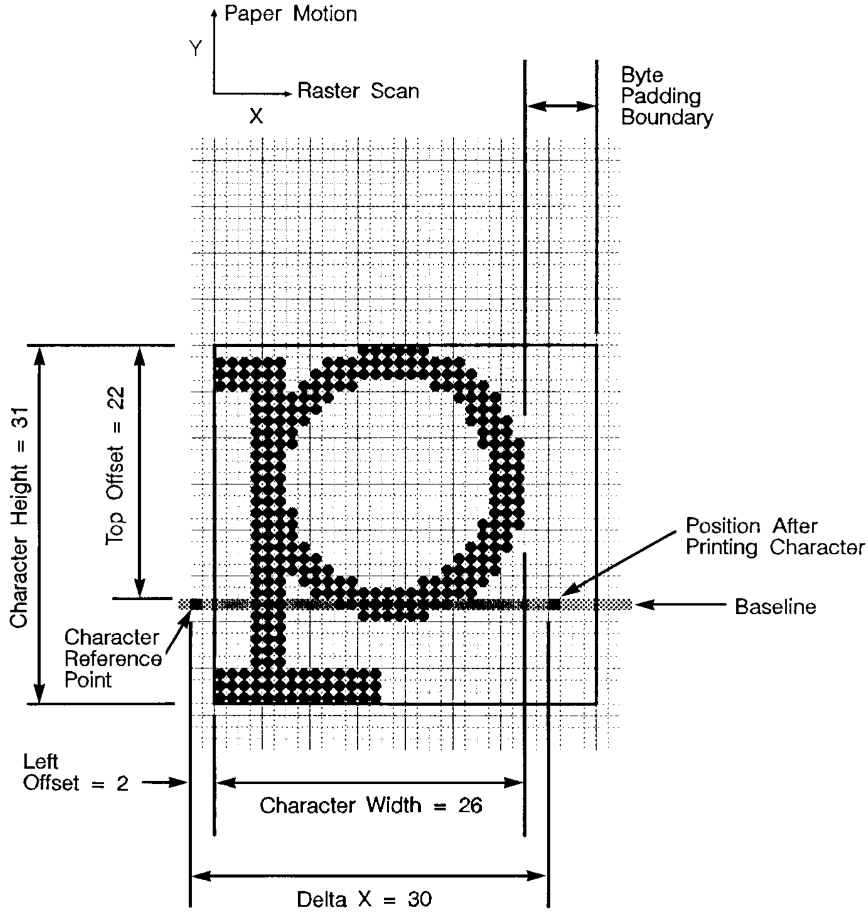
Delta X (SI)

Delta X specifies the number of quarter dots (radix dots) by which the horizontal position within the logical page coordinate system is incremented after printing the character. This value is only used by the printer when the font is proportionally spaced.

The legal range for delta X is –32768 to 32767 quarter units.

Character Data

Character data is a string of bytes containing the dot-per-bit image of the character or a run-length encoding with line repetition compressed format.



All values are in dots.

Figure 11-6 Portrait Character Example

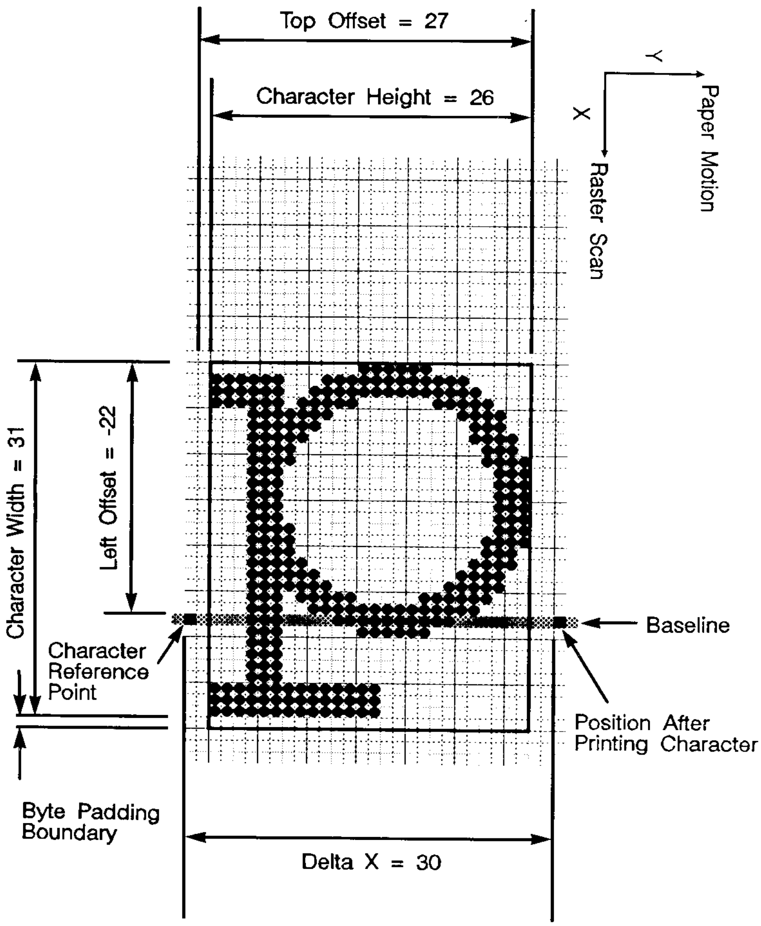


Figure 11-7 Landscape Character Example

Character Descriptor and Data Format for Intellifont Scalable Fonts

The character header contains a block of bytes that identify character outline data. Table 11-40 and Table 11-41 show the format of the Intellifont scalable character descriptor and data

.

Table 11-40 Intellifont Scalable Character Descriptor and Data Format

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (10)	Continuation (0)[1]
2	Descriptor Size	Class (3)
4	Contour Character Data: (in bytes) ⋮ see Table 11-41 for Contour Character Data	
#–2	Reserved (0)	Checksum[2]

1. Continuation is supported for classes 1, 2, 3 and 15 only.
2. These bytes appear only on the last continuation.

Table 11-41 Intellifont Scalable Contour Data Format

Byte	15 (MSB) 8	7 (LSB) 0
4	Contour Data Size	
6	Metric Data Offset	
8	Character Intellifont Data Offset	
10	Contour Tree Offset	
12	XY Data Offset	
14		
	Metric Data ⋮	
	Character Intellifont Data ⋮	

Table 11-41 Intellifont Scalable Contour Data Format

	Contour Tree Data ⋮
	XY Coordinate Data ⋮

Table 11-42 Intellifont Scalable Character Descriptors/Data Continuation Block

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (10)	Continuation (1)[1]
2	Contour Character Data, resumed: (in bytes, see Table 11-41. ⋮	
#–2	Reserved	Checksum[2]

1. Continuation is supported for Intellifont scalable fonts for class 3 only.
2. This byte appears only on the last continuation.

Table 11-43 Intellifont Scalable Compound Character Descriptor and Data Format

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (10)	Continuation (0)
2	Descriptor Size	Class (4)
4	Compound Character Escapement	
6	Number of Components	
8	Component List ⋮ see Table 11-46 for Component List Data	
–2	Reserved	Checksum

Format (UB)

This is the first byte of every character data block header. It specifies the format of the character descriptor and data. The format number for Intellifont scalable fonts is 10.

Table 11-44

Value	Format
4*	LaserJet Family (Raster)
10	Intellifont Scalable
15*	TrueType Scalable
* These are described elsewhere in this chapter.	

If the format number is different from that expected by the device, the character is discarded.

Continuation (B)

This is the second (and last) byte of every character data block header. It specifies whether the following data is the first (0) data block of a new character definition, or a continuation (1) block for a character definition which has already been received by the printer. Because the value field in a Character Definition command is limited to 32767 bytes, characters whose byte count exceed this must be sent in two or more blocks. Table 11-42 shows the continuation block for an Intellifont Scalable font.

Descriptor Size (UB)

This is the first byte of the character descriptor. It specifies the size of the character descriptor in bytes. The typical descriptor size for Intellifont scalable fonts is 2.

Class (UB)

Specifies the format of the character data. For Intellifont scalable fonts values 3 and 4 are used, as described below.

Table 11-45

Value	Class
1*	Bitmap
2*	Compressed Bitmap
3	Contour (Intellifont Scalable)
4	Compound Contour (Intellifont Scalable)
15*	TrueType Scalable
* These are described elsewhere in this chapter.	

Class 3 -Intellifont Scalable Character Contour Data

Class 3 is for Intellifont scalable contour character data. The contour character data is organized as described in Table 11-41. Bytes 0-3 contain the character descriptor.

Class 4 - Intellifont Scalable Compound Character Data

A class 4 character is a compound character and composition data follows. The composition data is organized as described in Table 11-43. The compound descriptor allows combining two different characters to produce a single compound character.

Contour Data Size (UI)

The size of the contour data including the size of this field. For a detailed description of this field, refer to *Intellifont Scalable Typeface Format*.

Metric Data Offset (SI)

The offset to the Metric Data relative to the address of the Contour Data Size field.

Character Intellifont Data Offset (SI)

The offset to the Character Intellifont Data relative to the address of the Contour Data Size field.

Contour Tree Offset (SI)

The offset to the contour Tree Data relative to the address of the Contour Data Size field.

XY Data Offset (SI)

The offset to the XY data relative to the address of the Contour Data Size field.

Metric Data

For information about Metric Data refer to the *Intellifont Scalable Typeface Format* document.

Character Intellifont Scalable Data

For information about Character Intellifont Scalable Data, refer to *Intellifont Scalable Typeface Format*.

Contour Tree Data

For information about Contour Tree Data, refer to *Intellifont Scalable Typeface Format*.

XY Coordinate Data

For information about XY Coordinate Data, refer to *Intellifont Scalable Typeface Format*.

Note For information on obtaining the *Intellifont Scalable Typeface Format* document, refer to *Related Documents* in the front of this manual.

Checksum

This is a checksum of all the contour character data. The checksum value is contained only in the last character data block.

Compound Character Escapement (SI)

The escapement in design units of a compound character.

Number of Components (UB)

The number of components of a compound character.

Component List

This is a list of component descriptions. The list contains Number of Components elements. Each component descriptor consists of 6 bytes as described in Table 11-46 below.

Table 11-46 Component Descriptor

Byte	15 (MSB) 8	7 (LSB) 0
0	Character Code	
2	X Offset	
4	Y Offset	

The Character Code is the character code number of a component of a compound character. X-offset is the offset of that component from the reference point (origin) in the x direction in design units. Y-offset is the offset in the y direction of a component from the reference point (origin) in design units.

Note The character code may be greater than the last code of the symbol set that is implied by the font type since a compound character can include components that are not part of the symbol set.

Character Descriptor and Data Format for TrueType Fonts

Table 11-47 shows the format of the TrueType character descriptor and data when a **continuation block is not required**. Table 11-49 (next page) shows the format of the TrueType character descriptor and data with **multiple character data blocks**.

Table 11-47 TrueType Character Descriptor (no continuation block required)

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (15)	Continuation (0)
2	Descriptor Size	Class (15)
4	*[additional descriptor data may be inserted here]*	
2 + Desc Size	Character Data Size	
4 + Desc Size	Glyph ID	
6 + Desc Size	TrueType Glyph Data ⋮	
# – 2	Reserved	Checksum
# = Character data block size as defined in Character Definition command.		

Table 11-48

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (15)	Continuation (0)
2	Descriptor Size	Class (15)
4	*[additional descriptor data may be inserted here]*	
2 + Desc Size	Character Data Size	
4 + Desc Size	Glyph ID	

Table 11-48 (continued)

6 + Desc Size	beginning of TrueType Glyph Data ⋮

Table 11-49 TrueType Character Descriptor (multiple character data blocks)

Byte	15 (MSB) 8	7 (LSB) 0
0	Format (15)	Continuation (1)
2	conclusion of TrueType Glyph Data ⋮	
# – 2	Reserved	Checksum
# = Character data block size as defined in Character Definition command.		

Format (UB)

This is the first byte of every character data block header. It specifies the format for character downloading. The number 15 designates the TrueType character format.

Table 11-50

Value	Format
4*	LaserJet Family (Raster)
10*	Intellifont Scalable
15	TrueType Scalable
* These are described elsewhere in this chapter.	

Continuation (B)

This is the second (and last) byte of every character data block header. It specifies whether the following data is the first (0) data block of a new character definition, or a continuation (1) block for a character definition which has already been received by the printer. Because the value field in a Character Definition command is limited to 32767 bytes, characters whose byte count exceed this must be sent in two or more blocks.

Descriptor Size (UB)

This is the first byte of the character descriptor. It specifies the size of the character descriptor in bytes. The character descriptor includes everything that is both after the continuation byte and prior to the Character Data Size field. The TrueType character descriptor includes this Descriptor Size byte plus the following Class byte. The minimum value for Descriptor Size is therefore 2. (Additional descriptor information, if any, can be added immediately after the Class byte.)

Class (UB)

This field is used to distinguish different character data types within a given character format. (For instance, among Intellifont characters of format 10, the Class byte is used to distinguish compound characters from simple Intellifont contour characters.) All TrueType scalable characters are handed to the TrueType font scaler in the same format, consequently, the Class byte does not provide vital new information. For TrueType, set the Class value to 15.

Table 11-51

Value	Class
1*	Bitmap
2*	Compressed Bitmap
3*	Contour (Intellifont Scalable)
4*	Compound Contour (Intellifont Scalable)
15	TrueType Scalable
* These are described elsewhere in this chapter.	

Character Data Size (UI)

The value of the Character Data Size should equal the sum of the sizes of the Character Data Size, Glyph ID, and TrueType Glyph Data fields. This value alerts the PCL interpreter when a continuation block is needed. The minimum possible value is 4. The value of Character Data Size plus Descriptor Size plus 4 (for the Format, Continuation, Reserved and Checksum bytes) will never be less than the value # given in the character download command. If the sum is exactly equal to #, then no continuation block is to be expected for the given character. However, if the sum exceeds #, then a continuation block is needed. A condition for the validity of a downloaded scalable TrueType character is that the sum of the # values for all of that character's data blocks equals the sum of the Descriptor Size and Character Data Size and 2 (for Reserved and Checksum), plus 2 times the number of character data blocks (for Format and Continuation bytes).

Glyph ID (UI)

This field is used by the TrueType font scaler as an ID number for the glyph data associated with the given character.

TrueType Glyph Data

This field contains the data segment associated with the given character as found in the **glyf** table of the original TrueType font file. See the description in *True Type Font Files*.

Checksum (UB)

The value of this byte, when added to the sum of all of the bytes in the Character Data Size, Glyph ID, and TrueType Glyph Data fields, should equal 0 in modulo 256 arithmetic. The Checksum is found only in the last character data block associated with a given character.

Character Definition Examples

Bitmap Portrait Character Example

To download a bitmap character descriptor and data for a portrait, 10 Pitch, 12 point, upright medium, Courier lower-case “p”, send:

$^{E}{}_{C}$***c112E** (112 is the decimal character code for an ASCII lower-case “p”)

$^{E}{}_{C}$**(s140W** [character descriptor and data]

Note Notice that the **140** appearing in the Character Definition Command accounts for 2 bytes of the Character Data Block Header, 14 bytes of Character Descriptor, and 124 bytes of Character Data. Since the Character Width is 26 dots, 4 bytes are needed per raster row. Also, since the Character Height is 31 dots, 124 bytes of Character Data for a Class 1 character is needed (4 × 31 = 124). No continuation block is to be expected.

Table 11-52 Character Format, Continuation, and Descriptor

FIELD NAME	VALUE	DESCRIPTION
Format	4	LaserJet Printer Family
Continuation	0	Not A Continuation Record
Descriptor Size:	14	Bitmap
Class:	1	Normal Raster
Orientation:	0	Portrait
Left Offset:	2	dots
Top Offset:	22	dots
Character Width:	26	dots
Character Height:	31	dots
Delta X:	120	Quarter Dots (30 Dots)

Table 11-53 Portrait Character Data Example

Dot Row	Bit Map	Decimal Equivalent			
01	00000000 00001111 11000000 00000000	0	15	192	0
02	11111100 01111111 11111000 00000000	252	127	249	0
03	11111100 11111111 11111100 00000000	252	255	252	0
04	11111101 11110000 00111110 00000000	253	240	62	0
05	00011111 11000000 00001111 00000000	31	192	15	0
06	00011111 10000000 00000111 00000000	31	128	7	0
07	00011111 00000000 00000111 10000000	31	0	7	128
08	00011110 00000000 00000011 10000000	30	0	3	128
09	00011110 00000000 00000011 11000000	30	0	3	192
10	00011100 00000000 00000001 11000000	28	0	1	192
11	00011100 00000000 00000001 11000000	28	0	1	192
12	00011100 00000000 00000001 11000000	28	0	1	192
13	00011100 00000000 00000001 11000000	28	0	1	192
14	00011100 00000000 00000001 11000000	28	0	1	192
15	00011110 00000000 00000001 11000000	30	0	1	192
16	00011110 00000000 00000011 11000000	30	0	3	192
17	00011110 00000000 00000011 10000000	30	0	3	128
18	00011111 00000000 00000111 10000000	31	0	7	128
19	00011111 10000000 00001111 00000000	31	128	15	0
20	00011111 11000000 00011111 00000000	31	192	31	0
21	00011101 11110000 01111110 00000000	29	240	126	0
22	00011100 11111111 11111100 00000000	28	255	252	0
23	00011100 00111111 11110000 00000000	28	63	240	0
24	00011100 00001111 11000000 00000000	28	15	192	0

Table 11-53 Portrait Character Data Example (continued)

25	00011100 00000000 00000000 00000000	28	0	0	0
26	00011100 00000000 00000000 00000000	28	0	0	0
27	00011100 00000000 00000000 00000000	28	0	0	0
28	00011100 00000000 00000000 00000000	28	0	0	0
29	11111111 11111100 00000000 00000000	255	252	0	0
30	11111111 11111100 00000000 00000000	255	252	0	0
31	11111111 11111100 00000000 00000000	255	252	0	0

Bitmap Landscape Character Example

To download the character descriptor and data for a landscape, 10 pitch, 12 point, upright, medium, Courier lower-case “p”, send:

$^{E}{}_{C}$***c112E** (112 is the decimal character code for an ASCII lower-case “p”)

$^{E}{}_{C}$**(s120W** [character descriptor and data]

Table 11-54 Character Format, Continuation and Descriptor

FIELD NAME	VALUE	DESCRIPTION
Format	4	LaserJet Printer Family
Continuation	0	Not A Continuation Record
Descriptor Size:	14	Bitmap
Class:	1	Normal Raster
Orientation:	1	Landscape
Left Offset:	-22	dots
Top Offset:	27	dots
Character Width:	31	dots
Character Height:	26	dots
Delta X:	120	Quarter Dots (30 dots)

Table 11-55 Landscape Character Data Example

Dot Row	Bit Map	Decimal Equivalent			
01	00000000 01111110 00000000 00000000	0	126	0	0
02	00000011 11111111 11000000 00000000	3	255	192	0
03	00001111 11111111 11110000 00000000	15	255	240	0
04	00011111 10000001 11111000 00000000	31	129	248	0
05	00111110 00000000 01111100 00000000	62	0	124	0
06	00111000 00000000 00111100 00000000	56	0	124	0
07	01111000 00000000 00011110 00000000	120	0	30	0
08	01110000 00000000 00001110 00000000	112	0	14	0
09	11100000 00000000 00001111 00000000	224	0	15	0
10	11100000 00000000 00000111 00000000	224	0	7	0
11	11100000 00000000 00000111 00000000	224	0	7	0
12	11100000 00000000 00000111 00000000	224	0	7	0
13	11100000 00000000 00000111 00001110	224	0	7	14
14	11100000 00000000 00000111 00001110	224	0	7	14
15	01110000 00000000 00001110 00001110	112	0	14	14
16	01110000 00000000 00001110 00001110	112	0	14	14
17	00111000 00000000 00011100 00001110	56	0	28	14
18	00111100 00000000 00111100 00001110	60	0	60	14
19	00011110 00000000 01111000 00001110	30	0	120	14
20	00001111 10000001 11110000 00001110	15	129	250	14
21	01111111 11111111 11111111 11111110	127	255	255	240
22	01111111 11111111 11111111 11111110	127	255	255	240
23	01111111 11111111 11111111 11111110	127	255	255	240
24	01110000 00000000 00000000 00001110	112	0	0	14

Table 11-55 Landscape Character Data Example (continued)

25	01110000 00000000 00000000 00001110	112	0	0	14
26	01110000 00000000 00000000 00001110	112	0	0	14

12 Macros

Introduction

A Macro is a group of PCL and H"P-GL/2 commands and/or data created by the user that is downloaded and stored in the printer. Once stored in the printer, a macro can be invoked upon request (using the assigned macro ID number), using a single command. When printing letters, for example, which include a company letterhead, the letterhead is repeated for each letter. This letterhead can be created as a macro and stored in the printer. Thus, whenever the letter is printed a macro command, sent to the printer, initiates the command sequence to print the letterhead.

Note HP-GL/2 commands are not supported within macros on all HP LaserJet printers. Refer to the "PCL Feature Support Matrix" in Chapter 1 of the *PCL 5 Comparison Guide* for specifics.

Macros eliminate the need to download the same information repeatedly, thus saving transmission time. However, the trade-off is that they consume user memory. If memory usage is a concern, a possible solution might be an HP custom macro cartridge.

Custom macros can be written and stored in ROM (read only memory), such as font-type cartridges or SIMM modules. When ROM-based macros are installed in the printer, they become available for selection. There is no need to download the macro and no user memory is consumed.

Hewlett-Packard provides a service to assist you by developing these custom products. For information contact:

Hewlett-Packard
Boise Printer Division
Attention: Product Specials
11311 Chinden Blvd.
Boise, ID 83714
(208) 323-3684

Macro Creation

Designate a unique identification (ID) number prior to the definition of a macro using the Macro ID command. This number is assigned to the macro. If a macro is already associated with this ID number, the previously existing macro is deleted from user memory during the definition of the new macro. Subsequent macro operations are accomplished using the macro ID number.

A macro is created in the printer in several steps. First, the start macro command is sent to the printer to indicate the start of a macro definition. Next, the printer commands, control codes, and data that constitute the macro are sent to the printer, in the intended order of their execution. Finally, the End Macro command is sent to the printer to indicate the end of the macro definition

Table 12-1

Assign ID Number	Macro ID command ($^{E}_{C}$&f14Y) assigns a unique identification number (14) to the macro.
Start Macro Definition	Macro Control command, $^{E}_{C}$&f0X, indicates the following commands and data are to be stored as a macro.
Macro data • • • Macro data	Escape sequences, control codes, HP-GL/2 commands and text required to perform the desired operation.
Stop MacroDefinition	Macro Control command, $^{E}_{C}$&f1X, identifies the end of the macro data (definition).

To invoke a macro, send the Macro ID command with the ID number of the macro, then send the Macro Control command with a value field of 2, 3, or 4 ($^{E}_{C}$&f2X, $^{E}_{C}$&f3X, or $^{E}_{C}$&f4X). This performs the macro in the specified mode: Execute (value field=2), Call (value field=3), or Overlay (value field=4). Refer to the following section “Macro Invocation” for a description of these modes.

Macros occupy a portion of user memory. The number of macros that can be stored simultaneously in user memory is limited only by the amount of available user memory.

Macro Invocation

There are three ways to invoke a macro: **execute, call,** and **overlay,** using the Macro Control command.

When a macro is **executed,** it begins performing its commands using the current modified print environment. Changes made to feature settings during macro execution are recorded in the modified print environment; these changes are retained upon completion of the macro execution.

When a macro is **called,** it begins performing its commands using the current modified print environment. Before the macro is performed, the current modified print environment is saved. Changes made to feature settings during a macro call are recorded in the modified print environment; however, these changes are not retained upon completion of the macro call. The modified print environment that existed prior to the macro call is restored.

When a macro is enabled for automatic **overlay,** its execution is the final operation each time a page is printed. Before the macro is performed, the current modified print environment is saved and replaced with the overlay environment. The overlay environment is a combination of the user default and the current modified print environments. Changes, made to feature settings during macro overlay, are recorded in the modified print environment; however, these changes are not retained upon completion of the macro overlay. The modified print environment that existed prior to the macro overlay is restored.

Note HP-GL/2 commands are not supported within macros on all HP LaserJet printers. Refer to the "PCL Feature Support Matrix" in Chapter 1 of the *PCL 5 Comparison Guide* for specifics.

The overlay environment consists of the current settings for the following features with the remainder of the environmental features set to their user default values:

Table 12-2

PCL CONTEXT:	
Page length	**Paper source**
Page size	**Number of copies**
Orientation	**Cursor position stack**
Registration	
HP-GL/2 CONTEXT:	
Current Pen Position	**Scaling Points**
Hard Clip Limits	**Soft Clip Window**
Logical Page Boundaries	**Fill Attributes**
Line Attributes	

Refer to Chapter 3 for descriptions of environments.

Note The current active position (CAP, or cursor position) is not part of the modified print environment. Therefore, the cursor position is not saved when a macro is called, nor is it restored upon completion. The Push/Pop Cursor Position command can be used to save and recall a cursor position.

Temporary / Permanent Macros

During its definition, a macro is automatically designated as temporary. A temporary macro is deleted from user memory during a printer reset. A macro can be designated as permanent to prevent the printer from deleting it during a printer reset. A macro is designated as temporary or permanent by reference to its ID number, using the Macro Control command described later in this chapter.

Note Temporary and permanent macros are removed from user memory whenever the printer's power is turned off.

Deleting Macros

There are several mechanisms provided by PCL macro control for explicit deletion of macros from user memory. These include commands to delete all macros, all temporary macros, or an individual macro by reference to its macro ID number, using the Macro Control command described later in this chapter.

Both temporary and permanent macros are deleted from memory whenever the printer's power is turned off.

Macro ID

The Macro ID command specifies an ID number for use in subsequent macro commands.

$^{E}{}_{C}$ **& f # Y**

=Macro ID number

Default **=** 0
Range **=** 0 - 32767

This number is used in subsequent macro operations.

The factory default macro ID is 0.

Example

To establish a macro ID number of 5, send:

$^{E}{}_{C}$ **& f 5 Y**

Macro Control

The macro control command provides mechanisms for definition, invocation, and deletion of macros.

$^{E}{}_{C}$ **& f # X**

Table 12-3

# =	0 - Start macro definition (last ID specified)
	1 - Stop macro definition
	2 - Execute macro (last ID specified)
	3 - Call macro (last ID specified)
	4 - Enable macro for automatic overlay (last ID specified)
	5 - Disable automatic overlay
	6 - Delete all macros
	7 - Delete all temporary macros
	8 - Delete macro (last ID specified)
	9 - Make macro temporary (last ID specified)
	10 - Make macro permanent (last ID specified)

Notes

A macro may call or execute another macro, which in turn may call or execute another macro; two levels of “nesting” are allowed.

Other than call and execute, no macro control operations may occur within a macro.

A printer reset command ($^{E}{}_{C}$E or UEL) is not allowed in a macro.

HP-GL/2 commands are not supported within macros on all HP LaserJet printers. Refer to the “PCL Feature Support Matrix” in Chapter 1 of the &cguide; for specifics. HP-GL/2 commands as well as the PCL command “Enter HP-GL/2 mode” ($^{E}{}_{C}$%#B) and the PCL picture frame directives are not recommended within a macro.

PCL macro control commands are not allowed while in HP-GL/2 context.

ROM-Based Macros

Macros can be supplied in ROM (read only memory), such as custom macro cartridges or on SIMM modules. The following considerations apply to all ROM-based macros.

- ROM-based macros include the macro ID numbers for their macros. These numbers cannot be changed. It is possible for a downloaded macro to be assigned the same ID number as a ROM macro. If this occurs, the downloaded macro has precedence. To access the ROM macro, the downloaded macro must be deleted.
- The ID numbers for ROM-based macros range from 0-32767.
- In case of conflicting ID numbers in ROM-based macros, the following priorities apply:

  1 Cartridge macros. (In printers that have two cartridge slots, one cartridge has priority over the other. This priority depends on the printer. Refer to Appendix E of the *PCL 5 Comparison Guide* for the macro cartridge priority.)

  2 SIMM macros. SIMM 1=highest, then SIMM 2, etc.

Example

To define a macro with an ID of 7, send:

$^{E}{}_{C}$**&f7y0X**

•

•

•

escape sequences, control codes, and data

•

•

•

To stop the macro definition, send:

$^{E}{}_{C}$**&f1X**

To make the macro with an ID of 7 permanent, send:

$^{E}{}_{C}$**&f7y10X**

To enable the macro with an ID of 7 for automatic overlay, send:

$^{E}{}_{C}$**&f7y4X**

To delete the macro with an ID of 7, send:

$^{E}{}_{C}$**&f7y8X**

Macro Control Example

The following illustrates the definition of a letterhead macro.

Table 12-4

$^{E}{}_{C}$**&f1Y**	Specify the Macro ID as one.
$^{E}{}_{C}$**&f0X**	Start Macro Definition.
$^{E}{}_{C}$**&a540h360V**	Position logo at (540, 360) decipoints in the PCL coordinate system.
$^{E}{}_{C}$***t150R**	Set graphics resolution to
150 dots-per-inch.	
$^{E}{}_{C}$***r1A**	Start raster image of logo.
$^{E}{}_{C}$***b60W [Raster data]**	Send the first raster line.
•	•
•	•
•	•
$^{E}{}_{C}$***b60W [Raster data]**	Send the last raster line.
$^{E}{}_{C}$***rC**	Stop raster graphics.
$^{E}{}_{C}$**&a540h780V**	Position for lettering at (540, 780) decipoints.
$^{E}{}_{C}$**(1X**	Select font with ID of 1.
ABC Corp.	Text
Post Office Box 15	Text

Table 12-4 (continued)

Fred, Texas 83707	Text
$^{E}{}_{C}$**&a540h960V**	Position first rule at (540, 960) decipoints**.**
$^{E}{}_{C}$***c10v4680H**	Set rule height and width.
$^{E}{}_{C}$***c0P**	Print the first rule.
$^{E}{}_{C}$**&a540h980V**	Position second rule at (540, 980) decipoint**s.**
$^{E}{}_{C}$***c&zero;P**	Print second rule.
$^{E}{}_{C}$**&a54&zero;h12&zero;&zero;V**	Position for first line of text at (540, 1200) decipoints.
$^{E}{}_{C}$**&f1X**	Stop Macro Definition.

This macro now can be **executed**, **called,** or **enabled** for automatic **overlay**.

13 The PCL Print Model

Introduction

The Print Model feature allows images and characters to be filled with any of the printer's predefined shading or cross-hatch patterns, or with a user-defined pattern. Images include any raster graphic, such as one created with PCL raster graphics commands (as described in Chapter 15, *Raster Graphics*); a rectangular fill area (as described in Chapter 14, *PCL Rectangular Area Fill Graphics*); or a character or characters selected from any font.

Print model operation defines a **pattern, source image,** and **destination image**. These images are applied to each other using the print model's transparent and opaque modes to produce a resulting image that is a combination of the others. The print model features, listed below, are illustrated in Figure 13-1 and Figure 13-2, and described on the following pages.

- Pattern
- Source Image
- Destination Image
- Source Transparency Mode
- Pattern Transparency Mode

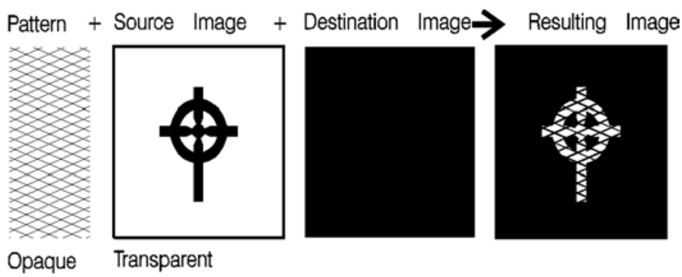


Figure 13-1 Print Model Imaging

Pattern

The design which is "painted" through the black ("1" bits) area of the source image onto the destination image. For patterns, the Print Model uses one of the printer's internal predefined eight shading patterns (see Figure 13-4) or one of the six cross-hatch patterns (see Figure 13-5), or a user-defined pattern.

Notes

When printing a page, text and raster images are printed using the **current pattern**. The default current pattern is 100% black. The print model allows the current pattern to be changed to white, to one of the six predefined cross-hatch patterns, to one of the eight shading patterns, or to a user-defined pattern. Once the current pattern is changed, it stays in effect until another is selected or the printer is reset. A reset returns the current pattern to its default value (100% black).

The current pattern does not always apply to rectangular area fill, which uses patterns defined by the rectangular area fill pattern commands. Refer to "Transparency Mode and Rectangular Area Fills" at the end of this chapter for additional information.

Source Image

An image in which the black ("1" bits) are replaced by the specified pattern. This is like a stencil through which the pattern is applied to the destination image. The source image may be defined as a rectangular fill area, a raster graphics image, or characters.

Destination Image

The image onto which the source image/pattern combination is placed. The destination image is the result of any previous operations.

Source Transparency Mode

The transparency or opaqueness of the source image's white pixels (the "0" bits) as they are applied to the destination image. Setting the source transparency mode to 1 (opaque) applies the source image's white pixels to the destination image; with a setting of 0 (transparent), these pixels have no effect on the destination.

Pattern Transparency Mode

The transparency or opaqueness of the white pixels in the pattern. When set to 0 (transparent), these pixels have no effect on the destination; when set to 1 (opaque), they are applied through the black pixels of the source pattern to the destination.



Figure 13-2 Opaque and Transparency Modes

Figure 13-3 illustrates the effects of the source and pattern transparency modes on the final image.

In the first example (1a), the transparency mode for both the source image and the pattern is transparent. Since the source mode is “transparent,” only the black-pixeled region (the circle) of the source image is overlaid on the destination. Since the pattern mode is also transparent, the patterned source image is applied only to the white areas of the destination image.

In the second example (1b), the source mode is still “transparent,” but the pattern mode is “opaque” — so the pattern’s white pixels are applied to the destination. The resulting image shows the entire circle region visible and patterned.

In the third example (1c), the source mode is “opaque” and the pattern mode is transparent. Since the source mode is opaque, the entire source image (the circle and the surrounding square) appears overlaid onto the destination. The pattern, however, is allowed to pour through only onto the white-pixeled area of the destination. The circle is visible in the result, but only two opposing quarters appeared patterned.

In the fourth example (1d), both source and pattern modes are “opaque.” The entire source image is overlaid onto the destination, and the entire circle is patterned.

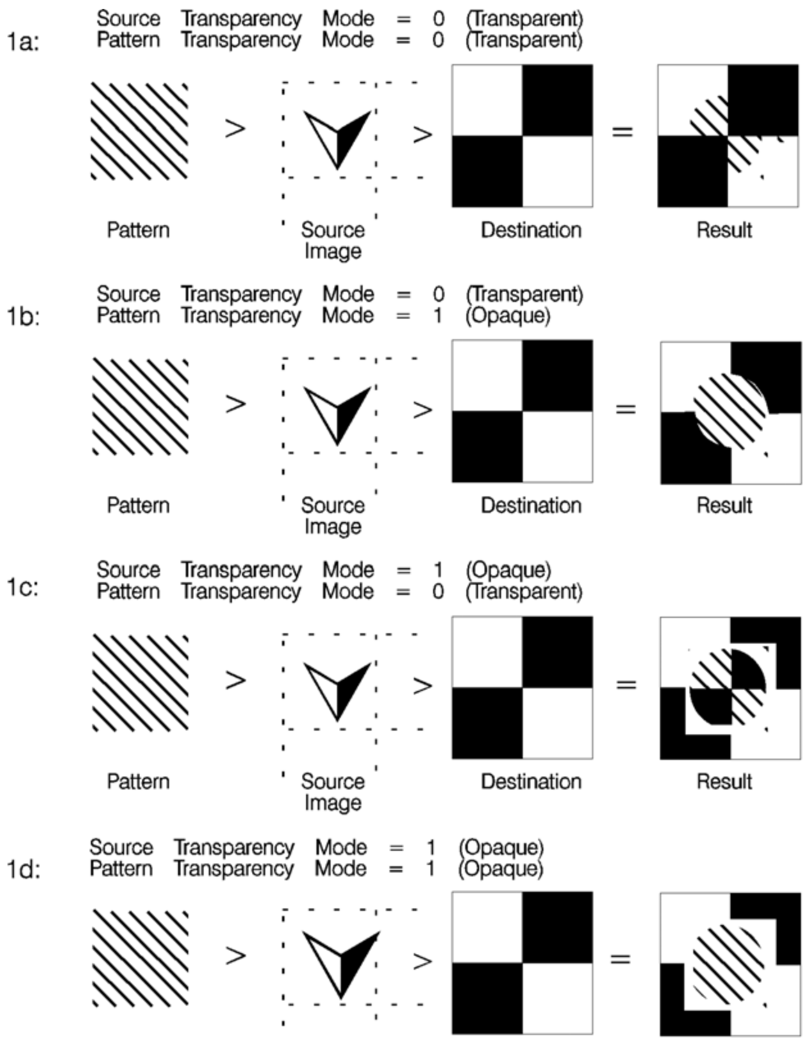


Figure 13-3 Effect of Transparency Modes on Images

Command Sequence

The following illustration shows the Print Model Command Sequence for selecting a current pattern and using it to fill a destination image.

Table 13-1

Operation	Comments
•	
•	
Download Page Data	Prior raster and character data downloaded to the page is considered destination image.
•	
•	
Select Transparency Modes	$^{E}{}_{C}$*v#N and/or $^{E}{}_{C}$*v#O
Select Specific Pattern ID	Pattern ID $^{E}{}_{C}$*c#G
and	
Select Pattern	$^{E}{}_{C}$*v#T (redefines current pattern)
Download Page Data (Source Image data)	Raster image/characters
Return to regular print mode	Default *current* pattern and
	transparency modes: $^{E}{}_{C}$*v0T (100% black pattern selected) and $^{E}{}_{C}$*v0N $^{E}{}_{C}$*v0O (transparency modes selected).
•	
•	
Download remaining page data	Transfer data for regular printing, or the above process may be repeated to produce another print model effect.

Table 13-1 (continued)

•	
•	
End of Page Data	

Source Transparency Mode Command

The Select Source Transparency Mode command sets the source image's transparency mode to transparent or opaque.

$^{E}{}_{C}$ * v # N

=0 - Transparent
1 - Opaque

Default **=** 0
Range **=** 0, 1 (other values cause the command to be ignored)

With a transparency mode of "0" (transparent), the white regions of the source image are not be copied onto the destination. With a transparency mode of "1" (opaque), the white pixels in the source are applied directly onto the destination.

Refer to the preceding definitions and the discussion of Figure 13-3 for an explanation of the effects of transparency.

Pattern Transparency Mode Command

The Pattern Transparency Mode command sets the pattern's transparency mode to transparent or opaque.

$^{E}{}_{C}$ * v # O

#	=0 - Transparent
	1 - Opaque

Default	**=**	0
Range	**=**	0, 1 (other values cause the command to be ignored)

A transparency mode of "0" (transparent) means that the white regions of the pattern image are not copied onto the destination. A transparency mode of "1" (opaque) means that the white pixels in the pattern are applied directly onto the destination.

Note When printing white rules, the pattern transparency is treated as if it were "opaque"; white rules erase black rules regardless of the transparency mode.

Refer to the preceding definitions and the discussion of Figure 13-3 and Figure 14-3 for an explanation of the effects of transparency.

Pattern ID (Area Fill ID) Command

The Pattern ID command (formerly called Area Fill ID) identifies the specific shading, cross-hatch, or user-defined pattern. (This command is also used for rectangular area fill. Refer to Chapter 14, *PCL Rectangular Area Fill Graphics.*)

$^{E}{}_{C}$ * c # G

Table 13-2

Selecting Shaded patterns		**Selecting Cross-Hatch patterns:**	
# =	1 thru 2 = 1-2% shade	# =	1 - Pattern #1
	3 thru 10 = 3-10% shade		2 - Pattern #2
	11 thru 20 = 11-20% shade		3 - Pattern #3
	21 thru 35 = 21-35% shade		4 - Pattern #4
	36 thru 55 = 36-55% shade		5 - Pattern #5
	56 thru 80 = 56-80% shade		6 - Pattern #6
	81 thru 99 = 81-99% shade		
	100 = 100% shade		
Selecting User-Defined patterns:[1]			
# = ID number of user-defined pattern			

1. Not supported on all LaserJet family printers. Refer to the “PCL Feature Support Matrix” in Chapter 1 of the *PCL 5 Comparison Guide* for specifics.

=ID number of user-defined pattern

Default = 0 (no pattern)
Range = 0 - 32767 (values outside the range are ignored)

For rectangular areas, the pattern material is determined by both the pattern ID and the value of the Fill Rectangular Area command. For other images, the pattern material is determined by the pattern ID and the value of the Select Pattern command.

Figure 13-4 and Figure 13-5 illustrate the HP-defined shading patterns and cross-hatched patterns, respectively.

Notes

This command is used for both the Select Pattern and Area Fill graphics (it is also described in Chapter 14, *PCL Rectangular Area Fill Graphics*). It is duplicated here for convenience.

For user-defined patterns, this command, sent prior to downloading a user-defined pattern, assigns an ID pattern number to the downloaded pattern. (For more information, see “User- Defined Graphics,” later in this chapter.)

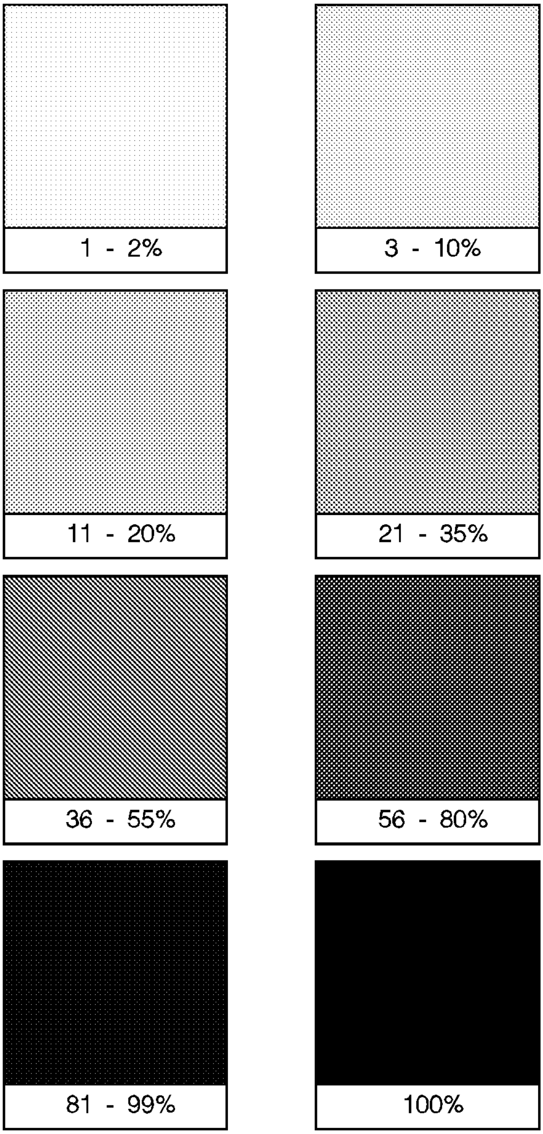


Figure 13-4 Shading Patterns

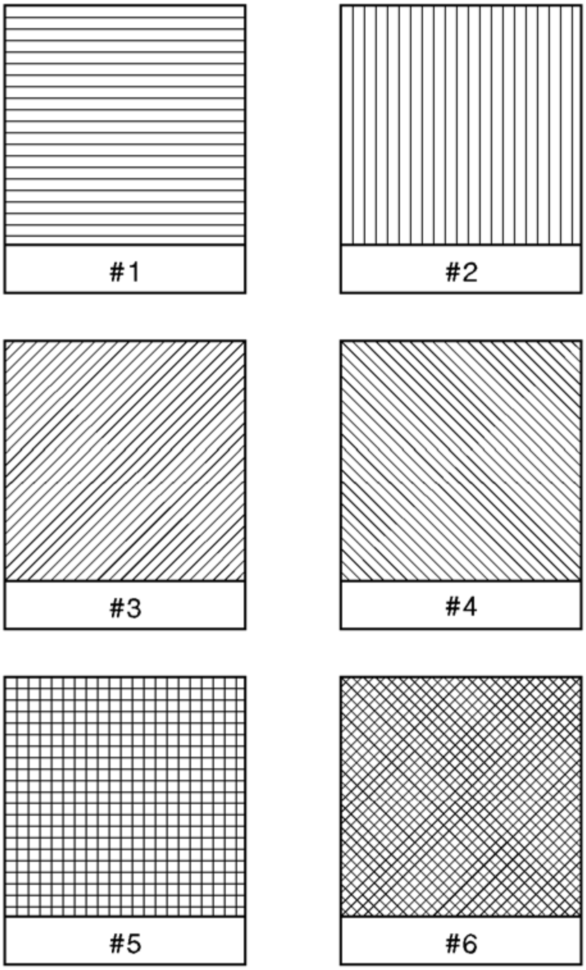


Figure 13-5 Cross-Hatch Patterns

Select Current Pattern Command

The Select Current Pattern command identifies the type of pattern to be applied onto the destination.

$^{E}{}_{C}$ * v # T

\# =0 - Solid black (default)
1 - Solid white
2 - Shading pattern
3 - Cross-hatch pattern
4 - User-defined pattern

Default **=** 0
Range **=** 0 - 4 (values outside of range are ignored)

This command selects which *type* of pattern is applied. For values 2, 3, and 4, the shading level (Figure 13-4), cross-hatch pattern (Figure 13-5), or user-defined pattern number is identified by the Pattern ID command described earlier in this chapter.

Notes

For selecting or changing the current pattern, the Select Current Pattern ($^{E}{}_{C}$*v#T) and the Pattern ID ($^{E}{}_{C}$*c#G) commands work together. **Sending the current pattern** (Select Current Pattern command) **alone does not change the current pattern; the Pattern ID must be sent first.** However, when selecting solid white (white rule) or solid black (black rule), only the Select Current Pattern command is required.

Once a current pattern is selected, that pattern applies to all images placed on the page until a new pattern is selected.

User-Defined Pattern Graphics

In addition to the eight shading patterns and six cross-hatch patterns, users can design their own patterns (area fill). These **user-defined patterns** are downloaded to the printer and controlled using three new commands:

- User-Defined Pattern $^{E}{}_{C}$*c#W [data]
- Set Pattern Reference Point $^{E}{}_{C}$*p#R
- Pattern Control $^{E}{}_{C}$*p#Q <Unknown> <list> >

User-Defined Pattern Implementation

To create a user pattern, a user defines a binary raster data image as a base pattern. This base pattern is downloaded to the printer using the User-Defined Pattern command. Prior to downloading the pattern, a Pattern ID command is sent to assign the user pattern an ID number. This ID number is used to select the pattern for printing and for pattern management.

To apply the pattern to an image, the printer duplicates or tiles (like placing ceramic tiles) the pattern across and down the page. This pattern can be applied to any image, or used as rectangular area fill.

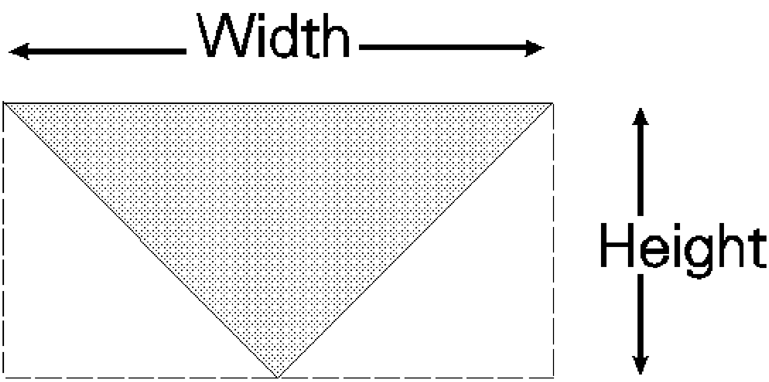


Figure 13-6 User-Defined Base Pattern Example

A user-defined pattern may be applied to any image in the same manner as the internal (cross-hatch or shade) patterns.

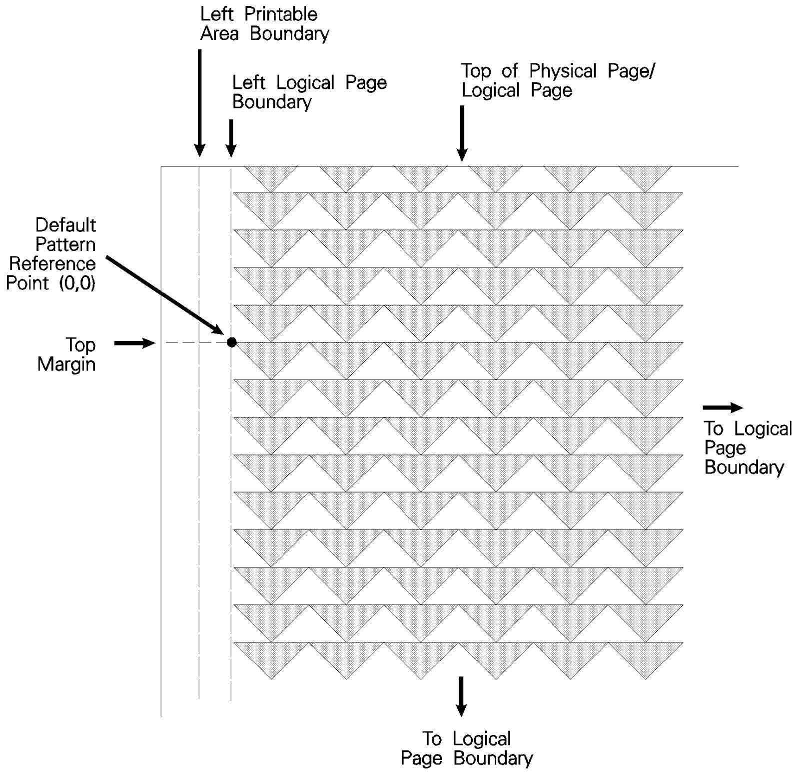


Figure 13-7 Pattern Layout Across the Printable Area

Pattern Reference Point

The pattern reference point is a position on the logical page at which the base pattern is positioned for tiling. The upper left corner of the base pattern is positioned at this point (see). The default pattern reference point is position 0,0. However, it is possible to set the pattern reference point to the current cursor position. This allows the pattern to be positioned or adjusted for fill areas. The pattern reference point may be shifted more than once for as many fill areas as there are on a page (the area must be filled before the tile point is moved for the next fill area).

Figure 13-8 shows two areas filled with the pattern reference point fixed at the default (0,0) position. The lower portion of the illustration shows two areas in which the pattern reference point was moved to the upper left corner of each area and the area filled separately.

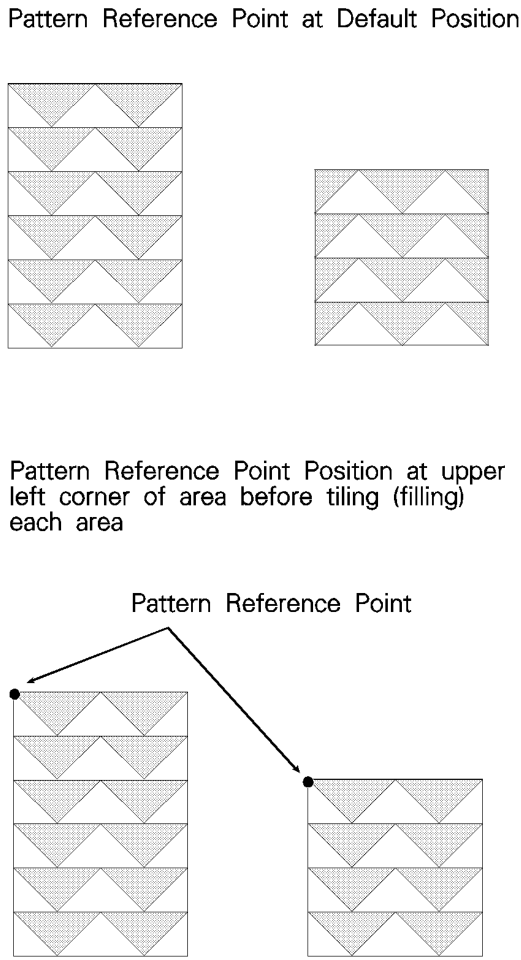


Figure 13-8 Moving Pattern Reference Point for Pattern Filling

User-Defined Pattern Command

The User-Defined Pattern command provides the means for downloading the binary pattern data that defines the user pattern.

^{E}C * c # W [pattern data]

=Number of pattern data bytes

Default = 0
Range = 0 - 32767 (values outside the range are ignored)

The value field (#) identifies the number of pattern data bytes that follow the User-Defined Pattern command. In addition to the binary pattern data, there are eight bytes of pattern descriptor (header) information included in this pattern data. The format for a 300 dpi resolution header is shown in Table 13-3, below.

Table 13-3 User-Defined Pattern Header (300 dpi resolution)

Byte	15 - MSB 8	7 LSB-0
0	Format (0)	Continuation (0)
2	Pixel Encoding (1)	Reserved (0)
4	Height in Pixels	
6	Width in Pixels	
8	Pattern image ⋮	

With the introduction of the LaserJet 4 printer, user-defined patterns can be printed either 300 or 600 dpi resolution. X Resolution and Y Resolution fields have been added to the header information included in the previous header. The format for the new header is shown in Table 13-4, below.

Table 13-4 Resolution-Specified User-Defined Pattern Header

Byte	15 - MSB 8	7 LSB-0
0	Format (0)	Continuation (0)
2	Pixel Encoding (1)	Reserved (0)

Table 13-4 Resolution-Specified User-Defined Pattern Header

4	Height in Pixels	
6	Width in Pixels	
8	X Resolution	
10	Y Resolution	
12	Pattern image ⋮	

Notes

The Master X and Master Y Resolution fields can be used to specify 600 dpi resolution for a pattern originally designed at 300 dpi. However, a pattern which was designed at 600 dpi is not available for selection at 300 dpi resolution.

Patterns are identified by some value (ID number). This is the current pattern ID number.

If the Pattern ID command is not used to assign an ID number to the user pattern, the existing (current) pattern ID value is used. If a pattern is already associated with the ID, that pattern is replaced with the new pattern.

Format (Byte 0)

This field, byte 0, must be set to “0.”

Continuation (Byte 1)

This field, byte 1, must be set to “0.” (This byte is for future printer support and does not currently provide any continuation operation.)

Pixel Encoding (Byte 2)

This field, byte 2, should be set to “1.” (This byte is reserved for future printer operation.)

Reserved (Byte 3)

This field, byte 3, is not currently used and must be set to 0.

Height in Pixels (Bytes 4 and 5)

This field, bytes 4 and 5, identifies the number of rows (height) of the pattern.

Width in Pixels (Bytes 6 and 7)

This field, bytes 6 and 7, identifies the number of pixels (width) of the pattern.

Pattern Image

This field contains the raster data for the pattern.

Master X Resolution (UI)

The Master X Resolution field is the pixel resolution in the X scan direction at which the pattern was designed.

Master Y Resolution (UI)

The Master Y Resolution field is the pixel resolution in the Y scan direction at which the pattern was designed.

User-defined Pattern Example

This example shows how the user-defined pattern command is used to create a user-defined patterns. For this example, a pattern of triangles is used. The first step is to design the base pattern triangle (in this case, using 64 bytes of data). The base pattern binary data is shown below:

```
11111111111111111111111111111111
01111111111111111111111111111110
00111111111111111111111111111100
00011111111111111111111111111000
00001111111111111111111111110000
00000111111111111111111111100000
00000011111111111111111111000000
00000001111111111111111110000000
00000000111111111111111100000000
00000000011111111111111000000000
00000000001111111111110000000000
00000000000111111111100000000000
00000000000011111111000000000000
00000000000001111110000000000000
00000000000000111100000000000000
00000000000000011000000000000000
```

This translates into the following 64 bytes in hexadecimal values:

Table 13-5

FF	FF	FF	FF
7F	FF	FF	FE
3F	FF	FF	FC
1F	FF	FF	F8
0F	FF	FF	F0
07	FF	FF	E0
03	FF	FF	C0
01	FF	FF	80
00	FF	FF	00
00	7F	FE	00
00	3F	FC	00
00	1F	F8	00
00	0F	F0	00
00	07	E0	00
00	03	C0	00
00	01	80	00

When using the 300 dpi User-Defined Pattern header (see Table 13-3). Set the eight bytes of header information to the following values:

Byte 0 - Format = 0 (00 hex)

Byte 1 - Continuation = 0 (00 hex)

Byte 2 - Pixel Encoding = 1 (01 hex)

Byte 3 - Reserved = 0 (00 hex)

Byte 4/5 - Height in Pixels = 0 / 16 (00 / 10 hex)

Byte 6/7 - Width in Pixels = 0 / 32 (00 / 20 hex)

Byte eight begins the first bytes of binary data.

The PCL code below downloads the user-defined pattern and assigns it an ID number of 3.

1 Specify the pattern ID number:

$^{E}{}_{C}$*c3G

Assigns an ID number of 3 to the pattern data which follows.

2 Send the User-defined Pattern command:

$^{E}{}_{C}$*c72W

Specifies that 72 bytes are to follow (8 bytes for header plus 64 bytes of pattern data).

3 Send the pattern header and binary data:

Table 13-6

```
00 00 01 00 00 10 00 20
FF FF FF FF
7F FF FF FE
3F FF FF FC
1F FF FF F8
0F FF FF F0
07 FF FF E0
03 FF FF C0
01 FF FF 80
00 FF FF 00
00 7F FE 00
00 3F FC 00
00 1F F8 00
00 0F F0 00
00 07 E0 00
00 03 C0 00
00 01 80 00
```

Notes

There must be an even number of bytes in user-defined pattern data, hence the trailing zeros (“padding”) in the last eight data rows above.

The user-defined pattern downloaded in the previous example is printed within a rectangular area in Chapter 14 under “Rectangular Area Fill Examples.”

In the previous example, the raster data code is presented in hexadecimal, however, the numbers in the escape sequences are decimal.

Set Pattern Reference Point Command

The Set Pattern Reference Point command causes the printer to tile patterns with respect to the current cursor position. This command also specifies whether the pattern rotates with the print direction or remains fixed.

$^{E}{}_{C}$ * p # R

=0 - Rotate patterns with print direction
1 - Keep patterns fixed

Default **=** 0
Range **=** 0,1 (values outside the range are ignored)

A value field of 0 rotates the patterns with changes in the print direction (see Print Direction command). For a value field of 1, patterns remain fixed for changes in print direction.

The default pattern reference point is the upper left corner of the logical page at the top margin (position 0,0). If the Set Pattern Reference Point command is not set, the pattern is tiled with respect to the default reference point.

Notes

All patterns are rotated for changes in orientation (refer to “Logical Page Orientation Command” in Chapter 5 of this manual).

This command applies to user-defined, shading, and cross-hatch patterns.

Pattern Control Command

The Pattern Control command provides a means for manipulating user-defined patterns.

$^{E}{}_{C}$ * c # Q

\# =0 - Delete all patterns (temporary & permanent)
1 - Delete all temporary patterns
2 - Delete pattern (last ID # specified)
3 - Reserved
4 - Make pattern temporary (last ID # specified)
5 - Make pattern permanent (last ID # specified)

Default **=** 0
Range **=** 0 - 5 (values outside the range are ignored)

For value fields 2, 4, and 5, the Pattern ID ($^{E}{}_{C}$***c#G**) command is sent prior to the Pattern Control command to identify the specific pattern for the Pattern Control command action.

14 PCL Rectangular Area Fill Graphics

Introduction

The PCL language includes commands for filling or shading rectangular areas on the page with pre-defined patterns, and allows creation and use of user-defined patterns. Pre-defined patterns include eight shading patterns and six cross-hatch patterns.

Note User-defined patterns are not supported on all LaserJet family printers. Refer to the “PCL Feature Support Matrix” in the *PCL 5 Comparison Guide*.

The first step in filling an area, if using a user-defined pattern, is to download the pattern (see Chapter 13). Next, position the cursor (using cursor move commands) and specify the dimension of the area (using the horizontal and vertical rectangle size commands). The cursor identifies the upper left corner of the rectangular fill area. Once the rectangle size and position are identified, select the specific pattern (**shading level** (Figure 14-1), **cross-hatch pattern** (Figure 14-2), or **user-defined pattern**), to be used for filling the rectangular area. Finally the command is issued to fill the defined rectangular area. Once a user-defined pattern has been downloaded, the fill procedure can be repeated as often as required.

Rectangular Area Fill Procedure

1 *For user-defined patterns:* if you have not done so already, download the binary pattern data (User-Defined Pattern Command - $^{E}{}_{C}$***c#W** - Chapter 13).

2 Position the cursor (choice of various cursor commands - Chapter 6).

3 Specify width of rectangle (Horizontal Rectangle Size Command - Decipoints=${}^{E}{}_{C}$*c#H, or PCL Units=${}^{E}{}_{C}$*c#A).

4 Specify height of rectangle (Vertical Rectangle Size Command - Decipoints=${}^{E}{}_{C}$*c#V, or PCL Units=${}^{E}{}_{C}$*c#B).

5 Select specific shade, cross-hatch, or user-defined pattern (Pattern ID Command - ${}^{E}{}_{C}$*c#G - Chapter 13)[1].

6 Fill rectangular area with pattern (Fill Rectangular Area Command - ${}^{E}{}_{C}$c#P).[1] This ends the procedure and prints the patterned area.

Note An area's width extends in the positive X-direction of the PCL coordinate system, and the height extends in the positive Y-direction.

The Pattern Transparency Mode controls how a pattern fills a rectangular area. Pattern Transparency Mode determines what effect transparent or opaque) the white pixels of the pattern have on the rectangular area (refer to "Pattern Transparency Mode Command" in Chapter 13).

1. White (1), black (0) or current (5) pattern also can be specified using this command.

Horizontal Rectangle Size (Decipoints) Command

This Horizontal Rectangle Size command specifies the rectangle width in decipoints.

$^{E}{}_{C}$ * c # H

\#   =Number of decipoints (1/720 inch)

Default **=** 0
Range **=** 0 - 32767 (valid to 4 decimal places)

The printer converts the specified width to printer dots by rounding up to an integral number of dots. For example, 5 decipoints, which corresponds to 2.08 dots in 300 dpi mode, is converted to 3 dots.

Horizontal Rectangle Size (PCL Units) Command

This Horizontal Rectangle Size command specifies the rectangle width in PCL Units.

$^{E}{}_{C}$ * c # A

\#   =Number of PCL Units

Default **=** 0
Range **=** 0 - 32767

For example, if the unit of measure is set to 300 units-per-inch, to specify a two-inch wide rectangle, send the command: $^{E}{}_{C}$***c600A**

The same command specifies a one-inch wide rectangle if the unit of measure is set to 600 units-per-inch.

Note The number of units-per-inch used in PCL dot moves is determined by the current setting of the **Unit of Measure** command (see “Unit of Measure Command” in Chapter 4).

Vertical Rectangle Size (Decipoints) Command

This Vertical Rectangle Size command specifies the rectangle height in decipoints.

$^{E}{}_{C}$ * c # V

\# =Number of decipoints (1/720 inch)

Default	**=**	0
Range	**=**	0 - 32767 valid to 4 decimal places

The printer converts the specified width to printer dots by rounding up to an integral number of dots. For example, 5 decipoints, which corresponds to 2.08 dots in 300 dpi mode, is converted to 3 dots.

Vertical Rectangle Size (PCL Units) Command

This Vertical Rectangle Size command specifies the rectangle height in PCL Units.

$^{E}{}_{C}$ * c # B</BLD

\# =Number of PCL Units

Default	**=**	0
Range	**=**	0 - 32767

For example, if the unit of measure is set to 300 units-per-inch, to specify a two-inch high rectangle, send the command: $^{E}{}_{C}$***c600A**

The same command specifies a one-inch high rectangle if the unit of measure is set to 600 units-per-inch.

Note The number of units-per-inch used in PCL dot moves is determined by the current setting of the **Unit of Measure** command (see "Unit of Measure Command" in Chapter 4).

Pattern ID (Area Fill ID) Command

The Pattern ID command (formerly called Area Fill ID) identifies the specific shading, cross-hatch, or user-defined pattern (see Figure 14-2) to be used when filling a rectangular area.

$^{E}{}_{C}$ * c # G

Table 14-1

Selecting Shaded Patterns		Selecting Cross-Hatch patterns:	
# =	1 thru 2 = 1- 2% shade	# =	1 - Pattern #1
	3 thru 10 = 3-10% shade		2 - Pattern #2
	11 thru 20 = 11-20% shade		3 - Pattern #3
	21 thru 35 = 21-35% shade		4 - Pattern #4
	36 thru 55 = 36-55% shade		5 - Pattern #5
	56 thru 80 = 56-80% shade		6 - Pattern #6
	81 thru 99 = 81-99% shade		
	100 = 100% shade		
Selecting User-Defined patterns:[1]			
# = ID number of user-defined pattern			

1. Not supported on all LaserJet family printers. Refer to the “PCL Feature Support Matrix” in Chapter 1 of the *PCL 5 Comparison Guide* for specifics.

Default **=** 0 (no pattern)
Range **=** 0 - 32767 (values outside the range are ignored)

The value field (#) identifies the level of shading, the cross-hatch pattern, or the user-defined pattern.

There are eight HP defined shading patterns defined within the PCL language. To specify one of the eight shading patterns, use any value within the value field range for the desired shade. For example, to select the 56-80% shade (shown in Figure 14-1) use a value of 56, or 80, or any value in between such as 73.

There are six HP defined cross-hatch patterns. To specify a cross-hatch pattern type, use a value between 1 and 6 to select a pattern as shown in Figure 14-2.

For user-defined patterns, this command, sent prior to downloading a user-defined pattern, assigns an ID pattern number to the downloaded pattern. (For more information, see “User-defined Pattern Graphics,” in Chapter 13.)

Note This command works in conjunction with the Fill Rectangular Area Command (described next in this section) and the Select Current Pattern Command (described in Chapter 13).

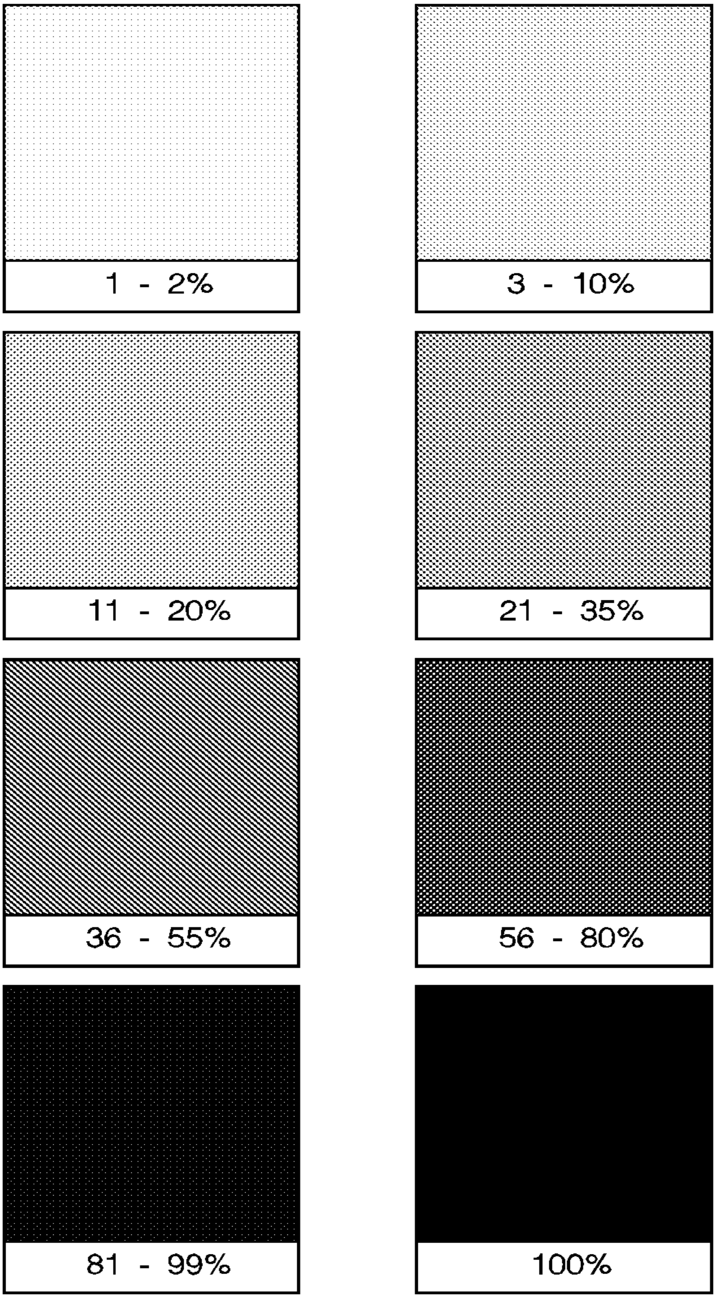


Figure 14-1 Shading Patterns

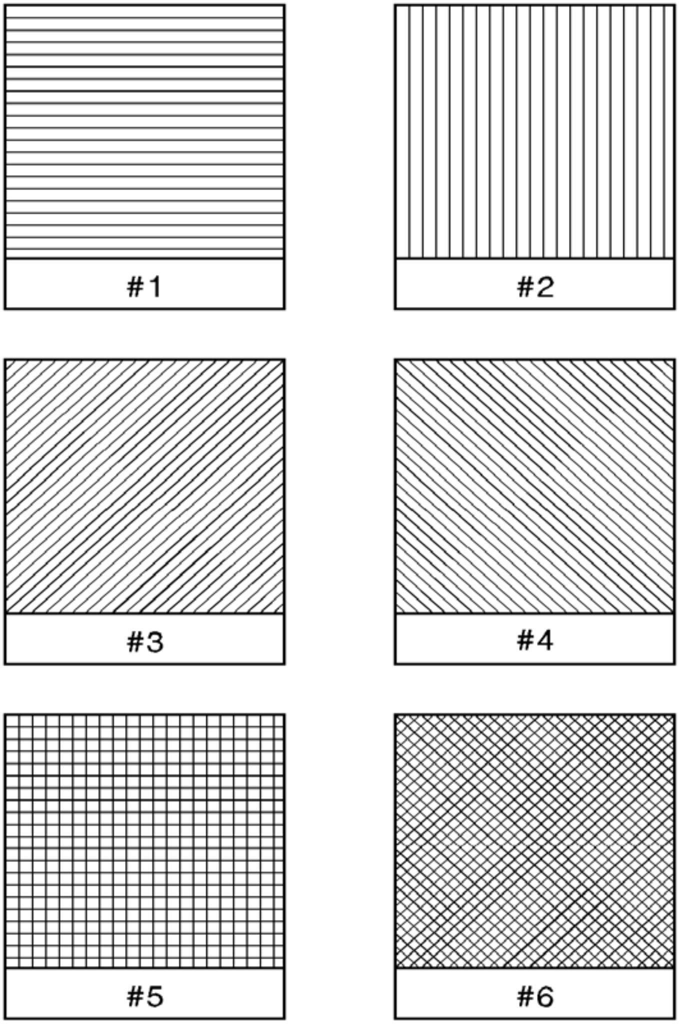


Figure 14-2 Cross-hatch Patterns

Fill Rectangular Area Command

This command fills (prints) a rectangular area of the specified width and height with the specified area fill.

$^{E}{}_{C}$ * c # P

#	=0 - Black fill (rule)
	1 - Erase (white) fill
	2 - Shaded fill
	3 - Cross-hatch fill
	4 - User-defined pattern fill
	5 - Current pattern fill

Default = 0
Range = 0 - 5 (values outside the range are ignored)

Black fill — fills the rectangular area with black fill.
White fill — erases any fill in the rectangular area (it fills the rectangular area with white fill).
Shaded fill — fills the rectangular area with one of the eight shading patterns as specified by the Pattern ID command.
Cross-Hatch fill — fills the rectangular area with one of the six cross-hatched patterns as specified by the Pattern ID command.
User-defined fill — fills the rectangular area with custom pattern data as specified by the Pattern ID command and downloaded by the User-Defined Pattern command.
Current Pattern — fills the rectangular area with the current pattern.

Notes

The order in which data (patterns/rules, text, raster) is received is the order in which it is processed during the rasterization of the page.

The current pattern is not applied to a rectangular area unless specified by this command.

The fill or pattern used as the current pattern is selected using the Select Current Pattern ($^{E}{}_{C}$*v#T) command. For a detailed description of the Select Current Pattern command refer to Chapter 13, *The PCL Print Model.*

Black fill (value field 0), also known as black rule, and the white fill (value field of 1) “patterns” do not have a choice of different patterns, and thus do not require a pattern specification using the Pattern ID command.

The upper left corner of the rectangular area is located at the cursor position when printing a rectangular area. After printing the rectangular area the cursor is returned to the upper left corner; the cursor position does not change positions as a result of printing a rectangular area.

Rectangular areas are independent of the text area and perforation skip mode; these boundaries are ignored (rectangles are not clipped at these boundaries). Addressable rectangular areas are limited to the logical page. Rectangular areas that extend outside the logical page are clipped at the logical page boundaries (refer to Figures 2-3 and 2-4 for logical page and printable area boundary specifications).

Transparency mode, described in Chapter 13, controls how the area fill pattern is applied to the page. Refer to the following section for a description of how transparency mode affects the rectangular fill area.

A white fill “erases” any data placed within the rectangular area prior to receipt of the white fill, regardless of the transparency mode settings. Data placed in a previously erased area is visible.

Pattern Transparency for Rectangular Area Fill

Pattern transparency, described in Chapter 13, under "Pattern Transparency Mode Command," affects how a pattern is applied to the rectangular fill area. Rectangular areas are special case images for transparency mode. The pattern and pattern type are selected by the Pattern ID command (EC*c#G) and the Fill Rectangular Area (EC*c#P) command (described earlier in this chapter).

Note Source transparency has **no effect** on the rectangular fill area since the rectangular area is viewed as all 1's (solid black) source image.

When applying a pattern (area fill) to the rectangular area, the usual transparency mode settings apply. The pattern transparency mode determines the effect white pixels of the pattern have on the destination for value fields 0 (black fill), 2 (shaded fill), 3 (cross-hatch fill), or 5 (current pattern fill) of the Fill Rectangular Area command.

The "0" bits of the area fill are either applied (opaque) or ignored (transparent) based on the transparency mode setting (see Figure 14-3). When a value field of 1 (white fill) is used, pattern transparency mode is **always treated as if it were opaque.**

The effect of transparency modes on rectangular areas is illustrated in Figure 14-3. In both examples, the source transparency mode is opaque regardless of the actual setting. In the first example, the pattern transparency mode is transparent; the white pixels in the pattern are not applied to the destination, so that the pattern is visible in only two quadrants of the destination. In the second example, the pattern transparency mode is opaque, and the pattern is visible in the entire rectangular area.

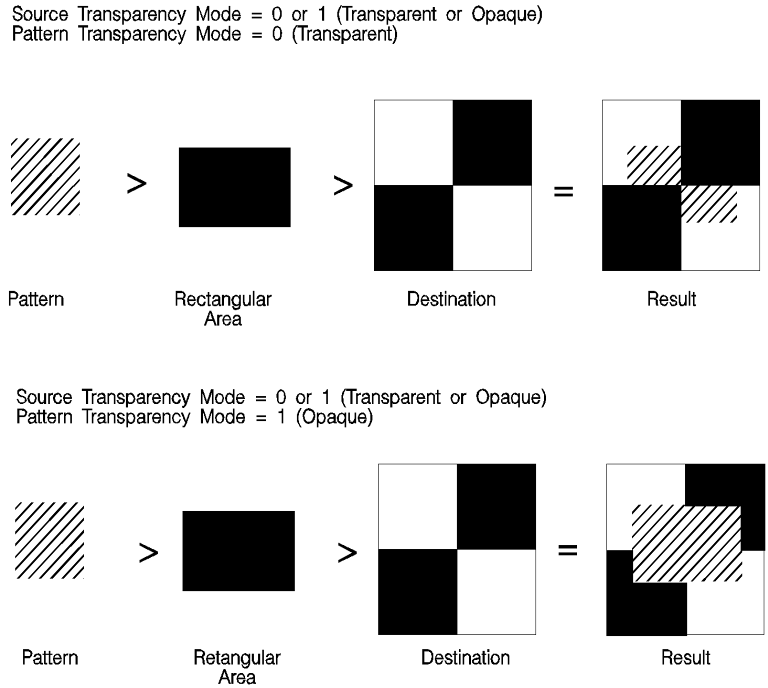


Figure 14-3 Effect of Transparency Modes on Rectangular Areas

Rectangular Area Fill Examples

This section shows example usage of area fill commands to print pre-defined patterns as well as user-defined patterns.

Pre-defined Pattern Examples

Solid Fill (Black/White)

To print a 900 by 1500 Unit black rule (3 inches by 5 inches at 300 units-per-inch), then white fill a small area inside the black rectangle, perform the following steps.

1 1. Position the cursor:

$^{E}{}_{C}$*p300x400Y

This moves the cursor to PCL Unit position (300, 400) within the PCL coordinate system.

2 2. Specify the width of the rule:

$^{E}{}_{C}$*c900A

This sets the rule width to 900 PCL Units (3 inches at 300 units-per-inch).

3 3. Specify the height of the rule:

$^{E}{}_{C}$*c1500B

This sets the rule height to 1500 PCL Units (5 inches at 300 units-per-inch).

4 4. Print the rule:

$^{E}{}_{C}$*c0P

This example prints a black filled rectangular area.

5 5. Position the cursor inside the rectangular area:

$^{E}{}_{C}$*p600x700Y

6 6. Specify the width and height for the smaller white fill rectangular area:

$^{E}{}_{C}$*c300a600B

7 7. Select the white fill and print.

$^{E}{}_{C}$*c1P

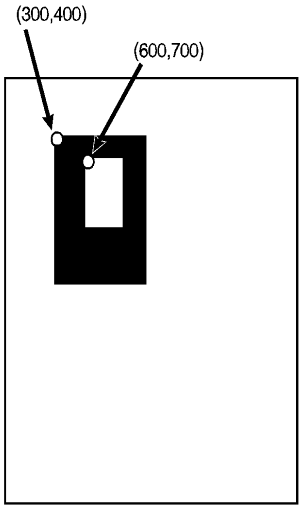


Figure 14-4 Solid Fill Example

Shaded Fill

To print a 900 by 1500 Unit 25% shaded rectangle (3 inches by 5 inches at 300 units-per-inch), perform the following steps.

1 Position the cursor:

$^{E}{}_{C}$*p300x400Y

This moves the cursor to PCL Unit position (300, 400) within the PCL coordinate system.

2 Specify the width of the rectangle:

$^{E}{}_{C}$*c900A

This sets the rectangle width to 900 PCL Units (3 Inches at 300 units-per-inch).

3 Specify the height of the rectangle:

$^{E}{}_{C}$*c1500B

This sets the rectangle to 1500 PCL Units (5 inches at 300 units-per-inch).

4 Specify the Pattern ID:

$^{E}{}_{C}$*c25G

This sets the Pattern ID to 25.

1 Print the rectangular shaded area:

$^{E}{}_{C}$*c2P

This example prints the following:

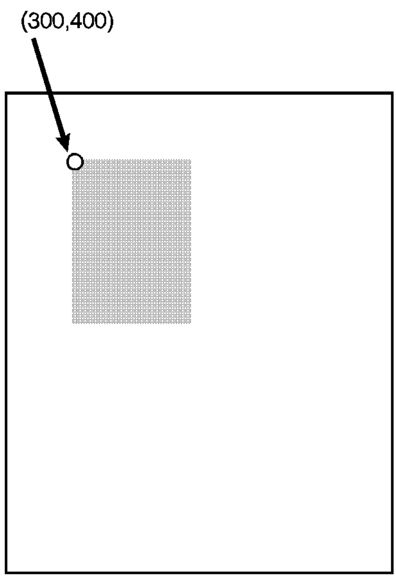


Figure 14-5 Shaded Fill Example

Cross-hatch Fill

To print a 900 by 1500 Unit rectangular area (3 inches by 5 inches at 300 units-per-inch), filled with a horizontal cross-hatch pattern, perform the following steps:

1. Position the cursor:

   $^{E}{}_{C}$*p300x400Y

Moves the cursor to PCL Unit position (300,400) within the PCL coordinate systems.

2. Specify the width of the rectangle:

   $^{E}{}_{C}$*c900A

Sets the rectangle width to 900 PCL Units (3 inches at 300 units-per-inch).

3. Specify the height of the rectangle:

   $^{E}{}_{C}$*c1500B

Sets the rectangle height to 1500 PCL Units (5 inches at 300 units-per-inch).

4. Specify the Pattern ID:

   $^{E}{}_{C}$*c1G

Sets the Pattern ID to 1.

5. Print the rectangular pattern-filled area:

   $^{E}{}_{C}$*c3P

This example prints the following:

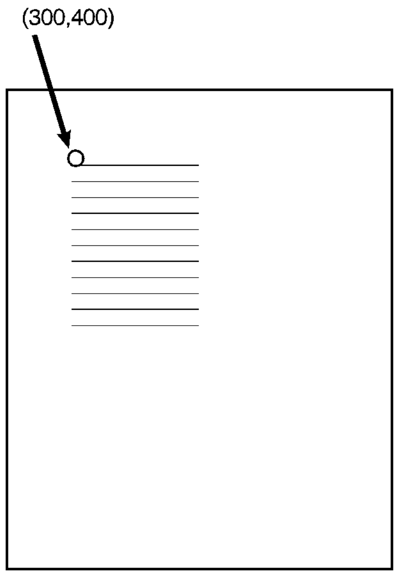


Figure 14-6 Patterned Fill Example

User-defined Pattern Example

This example shows how to print a user-defined pattern which has already been downloaded to the printer. For this example we will print the pattern of triangles which was downloaded in the example in Chapter 13, under “User-Defined Pattern Command.”

The following commands define a 600 PCL Unit square rectangular area, select ID number 3, and print the user-defined pattern associated with that ID number.

1 Position the cursor:

$^{E}{}_{C}$*p300x400Y

Moves the cursor to PCL Unit position (300,400) within the PCL coordinate systems.

2 Specify the width and height of the rectangle:

$^{E}{}_{C}$*c600a600B

Sets the rectangle width and height both to 600 PCL Units (2 x 2 inches at 300 Units/inch).

3 Specify the Pattern ID:

$^{E}{}_{C}$*c3G

Sets the Pattern ID to 3.

4 Print the user-defined pattern-filled area:

$^{E}{}_{C}$*c5P

This example prints as shown in Figure 14-7.

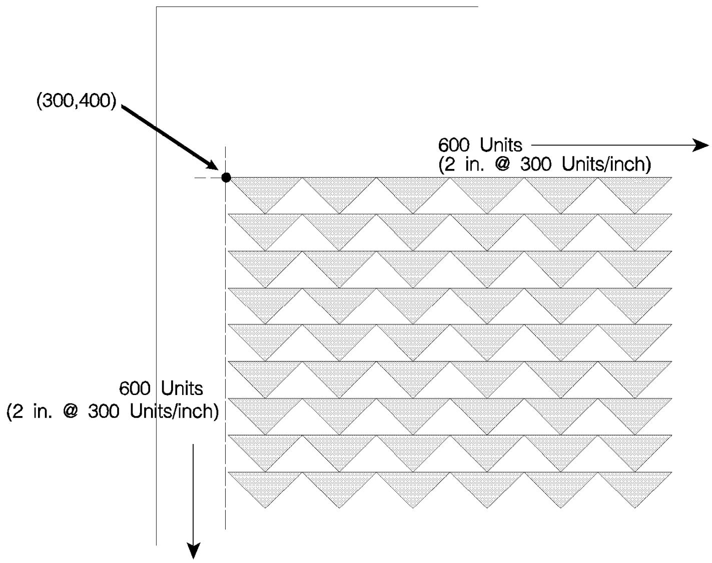


Figure 14-7 User-Defined Pattern Fill Example

15 Raster Graphics

Introduction

A raster image is an image composed of dots. Pictures in newspapers or on television screens (also, a page printed by this printer) are examples of raster images. The PCL language includes commands for printing raster graphic images. These commands enable the LaserJet printer to receive binary data and print it as a raster image.

The binary data used to create a raster image is divided into dot rows: a row describes a one-dot-high strip of the image. Each dot position within a row is represented by a binary data bit. If a bit in a row is set to one, a dot is printed; if the bit is set to zero, no dot is printed for that position. A dot row of raster image data is transferred to the printer as a string of bytes containing a dot-per-bit representation of the row.

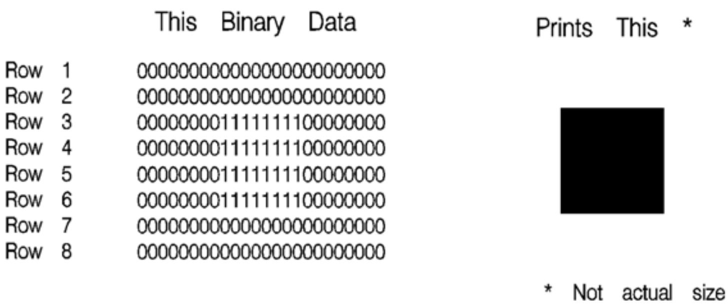


Figure 15-1 Binary Raster Data

Since it takes a considerable amount of data to create even a small raster image, several methods are provided to reduce the amount of data needed to define an image. (Note, that the above illustration creates a rectangle 0.013 by 0.027 inches; a binary "1" = 1 dot = 1/300 inch.) These reduction techniques include

several binary data compression methods, and additional reduction techniques associated with the **raster area** feature (see Figure 15-2).

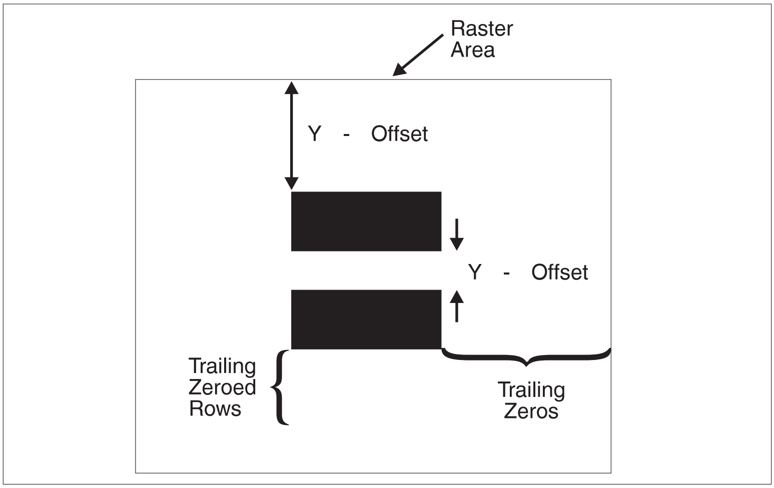


Figure 15-2 Raster Area

Data compression methods include: run-length encoding, tagged image file format (TIFF), delta row, and adaptive compression. These techniques are described in detail later in this section, under the Set Compression Method command.

In addition to the compression methods, the raster area feature provides some other raster reduction techniques which utilize a defined raster area. The raster area is defined by a width and height which are set using the Raster Width and the Raster Height commands.

Zeroed rows at the top and within the raster image can be eliminated by using the Y-offset feature. Y-Offset identifies how many rows to skip (zero fill). The Y-offset command specifies the Y-offset or number of rows for the printer to fill with zeroed rows. This provides a reduction in data for increased efficiency.

Trailing zeroed full rows at the end of the raster image need not be sent. The printer automatically fills in any unsent zeroed rows from the end of the raster image (last raster row with any “1”s) to the bottom of the raster area.

The final data reduction technique provided by the raster area involves the printer's ability to fill in trailing zeros to the edge of the raster area. Any zeros following the last “1“ in the raster row to the edge of the picture area need not be sent. The printer automatically fills them. This technique eliminates the need to transmit raster data rows that are all the same length, as required in a raster image which does not use the raster area feature.

The raster area represents a boundary. Within this boundary the printer zero-fills missing rows and fills in short rows to the edge of the raster area. However, in addition to filling to the boundaries of the raster area, the printer also clips any raster line which extends beyond the boundary. Thus, if an image extends beyond the raster area, then that portion of the image is not printed.

When the raster area reduction techniques are used in conjunction with the raster compression techniques, a considerable savings in data can be realized. This results in a saving of host storage and data transmission time. However, these reduction techniques do not reduce the amount of printer memory required for page formatting.

Raster Graphics Command Sequence

PCL raster commands include: Start Raster Graphics and End Raster Graphics commands, Transfer Raster Data by Row, Raster Compression, Raster Presentation, Raster Resolution, Raster Height and Raster Width (which define the raster area), and Raster Y Offset commands. The normal sequence of execution for these commands is shown below:

Table 15-1

Raster Presentation
Raster Resolution
Raster Height
Raster Width
Start Raster Graphics
Y Offset
Raster Compression
Transfer Raster Data
⋮
Transfer Raster Data
Y Offset
Transfer Raster Data
⋮
Y Offset
Raster Compression
Transfer Raster Data
⋮
Raster Compression
Transfer Raster Data
End Raster Graphics

The emphasis in the previous command sequence is that the Raster Presentation Mode, Raster Resolution, Raster Height, and Raster Width are all set outside the *start..data..end* sequence of commands. Also, the entire image is sent during the *start..data..end* sequence, choosing the most effective compression method for each raster row of data.

Raster Presentation, Raster Resolution, Raster Height, Raster Width, and Raster Compression are all true modes. Once specified, the printer remains in that mode unless explicitly changed by issuing the command again, or reset to default values by a soft reset, self test, font printout, or power cycle.

Note Only raster data appearing within the intersection of the logical page, the printable area, the raster width, and height is printed. If raster width and/or raster height have not been set (are defaulted), then the intersection of the logical page and the printable area determines where raster graphics appear; raster data is clipped to the printable area.

Raster Graphics Resolution Command

Raster graphics can be printed at various resolutions. This command designates the resolution of subsequent raster data transfers in dots-per inch.

$^{E}{}_{C}$ * t # R

\# = 75 - 75 dots-per-inch
100 - 100 dots-per-inch
150 - 150 dots-per-inch
200 - 200 dots-per-inch[1]
300 - 300 dots-per-inch
600 - 600 dots-per-inch

Default = 75
Range = 75, 100, 150, 200, 300, 600

This command must be sent prior to the start graphics command. The factory default resolution is 75 dots-per-inch.

Note Lower resolution graphics occupy less user memory. For example, the number of bits required to represent a two-inch by three-inch image at 75 dots-per-inch is 33,750. The same image at 300 dots-per-inch requires 540,000 bits.

When configured for 300 dpi resolution, the printer automatically expands raster graphics transferred at resolutions less than 300 dots-per-inch to 300 dots-per-inch during printing. illustrates how a single bit is translated into the corresponding printed dots in various graphics resolutions when the printer is configured for 300 dpi.

1. Only available if the printer is configured for resolution=600 dpi.

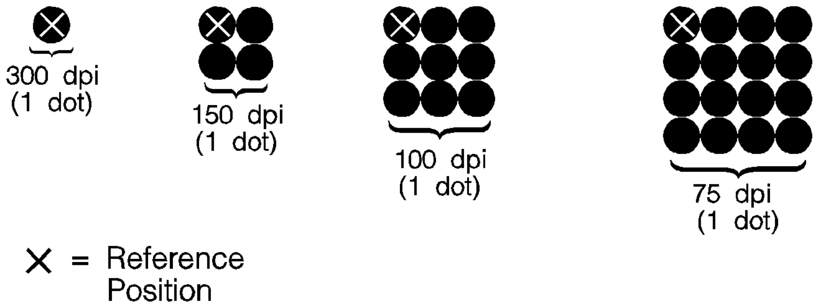


Figure 15-3 Raster Graphics Expansion - at 300 dpi

Note Rectangular area fills and character data are not affected by changes in resolution. Rectangular Area fills and character data always print at the maximum resolution, regardless of the resolution setting.

When configured for 600 dpi resolution, the printer automatically expands raster graphics transferred at resolutions less than 600 dots-per-inch to 600 dots-per-inch during printing. illustrates how a single bit is translated into the corresponding printed dots in various graphics resolutions when the printer is configured for 600 dpi.

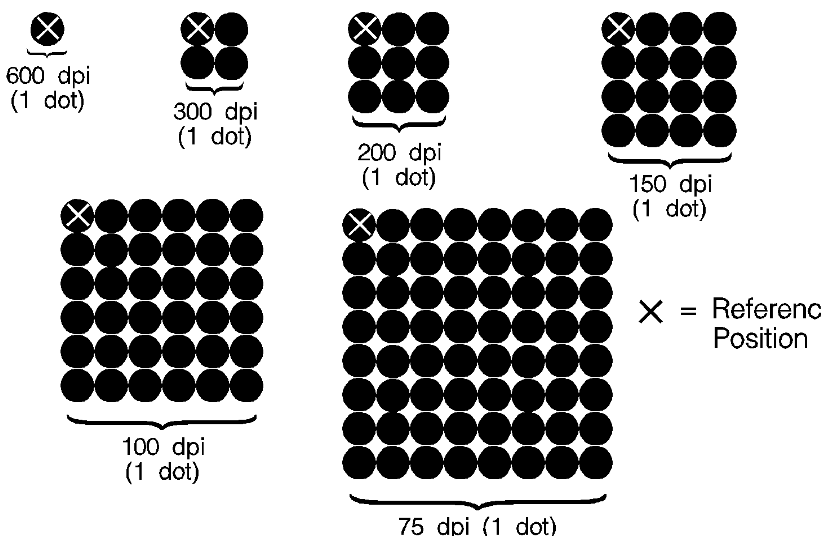


Figure 15-4 Raster Graphics Expansion - at 600 dpi

Raster Graphics Presentation Mode Command

The Raster Graphics Presentation command specifies the orientation of the raster image on the logical page.

E_C * r # F

\# =0 - Raster image prints in orientation of logical page
3 - Raster image prints along the width of the physical page

Default = 3
Range = 0, 3

A value of 0 indicates that a raster row ‘will be printed in the positive X-direction of the PCL coordinate system. (The print direction translates the PCL coordinate system.)

A value of 3 indicates that the raster graphics will be printed along the width of the physical page, regardless of logical page orientation. In portrait orientation, a raster row is printed in the positive X-direction of the PCL coordinate system and a subsequent raster row is printed beginning at the next dot row position in the positive Y-direction. In landscape orientation, a raster row is printed in the positive Y-direction of the PCL coordinate system and a subsequent raster row is printed beginning at the next dot row position in the negative X-direction. Figure 15-5 illustrates presentation mode 0 and 3.

Table 15-2

Raster Presentation Mode	Orientation	Default Graphics Margin
0	portrait	logical page left bound
0	reverse portrait	logical page left bound
0	landscape	logical page left bound
0	reverse landscape	logical page left bound
3	portrait	logical page left bound
3	reverse portrait	logical page left bound

Table 15-2 (continued)

3	landscape	50 dots in from the logical page *top* bound
3	reverse landscape	50 dots in from the logical page *top* bound

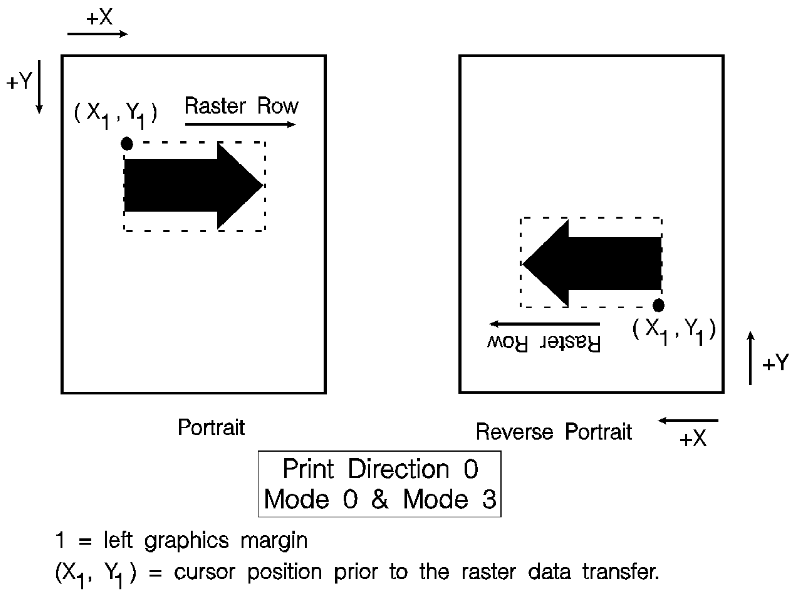
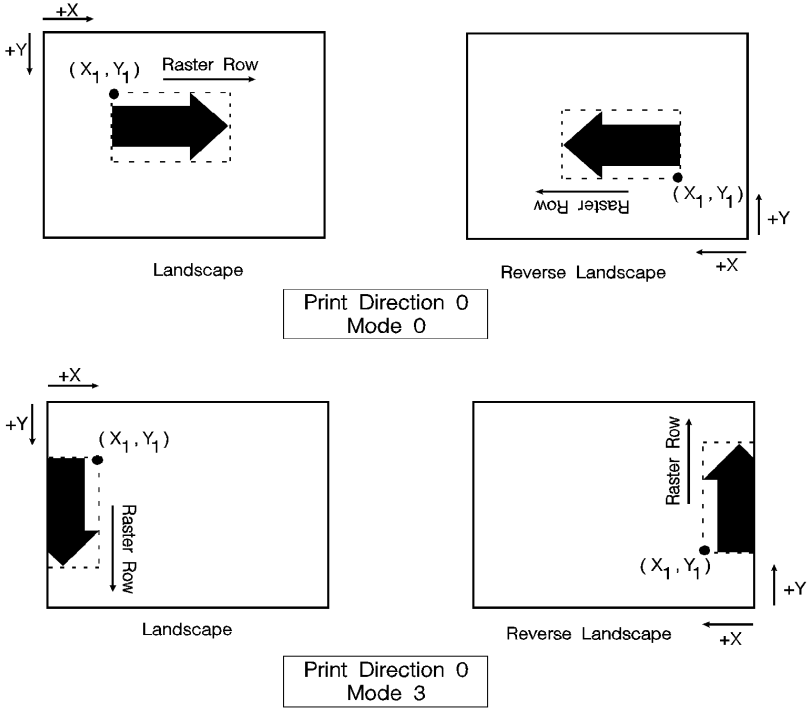


Figure 15-5 Raster Graphics Presentation Mode for Portrait Orientation



1 = left graphics margin

(X_1, Y_1) = cursor position prior to the raster data transfer.

Figure 15-6 Raster Graphics Presentation Mode for Landscape Orientation

Raster Height Command

The Raster Height command specifies the height in raster rows of the raster area. Height is the direction perpendicular to the direction that raster rows are laid down, hence, height is subject to the current raster presentation mode and print direction (see Figure 15-7).

$^{E}{}_{C}$ * r # T

\# = Height in raster rows

Default = N/A
Range = 0 to (logical page length – current Y-position of the 0,cursor)*

* Greater values default to (logical page length – current Y-position of the cursor)

This command fills the raster area to the full raster height with zeroed rows. Unspecified rows map to either white or transparent depending on the source transparency mode.

When a Transfer Raster Data command is received that causes any raster row to extend beyond the row boundary set by the Raster Height command, the row outside the boundary is clipped. This includes the case where the cursor is moved beyond the height boundary with a Raster Y Offset command and the printing of raster data is attempted.

If you have specified either a raster height or a raster width of 0 and a Start Raster Graphics (or Transfer Raster Data) command is received, then the entire raster graphic is clipped. If both a raster height and a raster width are specified (non-zero) and a Start Raster Graphics (or Transfer Raster Data) command is received then the raster area is guaranteed to be logically zeroed-out.

If the raster height is not set, the raster height is ignored so that no padding or clipping of rows takes place.

This command is ignored after the Start Raster Graphics or Transfer Raster Data commands until the next End Raster Graphics command.

Note Only raster data appearing within the intersection of the logical page, the printable area, and if set, the raster width and height is printed. Data outside the intersection is clipped.

Upon receiving an End Raster Graphics ($^E{}_C$***rC**) command, the cursor position is set to the left graphics margin of the next raster row after the raster height boundary.

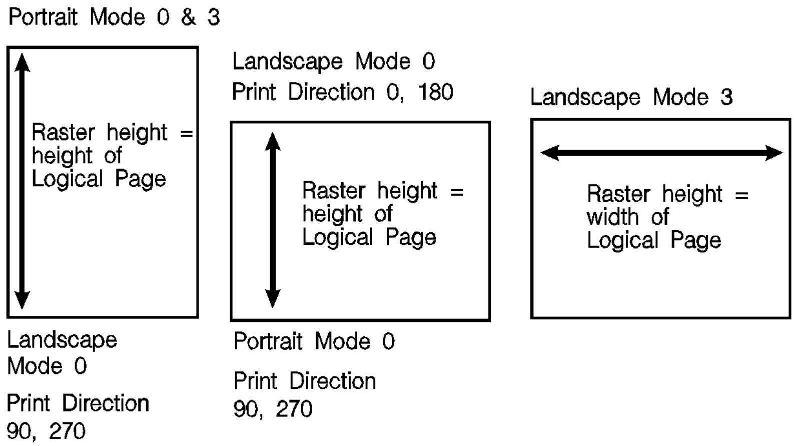


Figure 15-7 Maximum Raster Height

Raster Width Command

The Raster Width command specifies the width in pixels of the raster area. Width is in the direction that the raster rows are laid down, hence, width is subject to the current raster presentation mode and print direction (see Figure 15-8).

$^{E}{}_{C}$* r # S

=Width in pixels of the specified resolution

Default **=** depends on raster presentation mode - when presentation mode is 0, width = width of logical page - left graphics margin when presentation mode is 3 then width = dimension of logical page along paper length - left graphics margin

Range **=** 0 to (logical page width – left graphics margin)*

*Greater values default to the (logical page width – left graphics margin).

This command allows you to implicitly tell the printer to pad raster rows that are not specified for the full raster width with zeros. Unspecified data maps to either white or transparent depending on the source transparency mode.

When a Transfer Raster Data command is received that specifies a row of data that is longer than the raster width, the data that extends past the raster width is clipped.

This command is ignored after the Start Raster Graphics or Transfer Raster Data commands, until the next End Raster Graphics command.

Note Only raster data appearing within the intersection of the logical page, the printable area, and if set, the raster width and height is printed. Data outside the intersection is clipped.

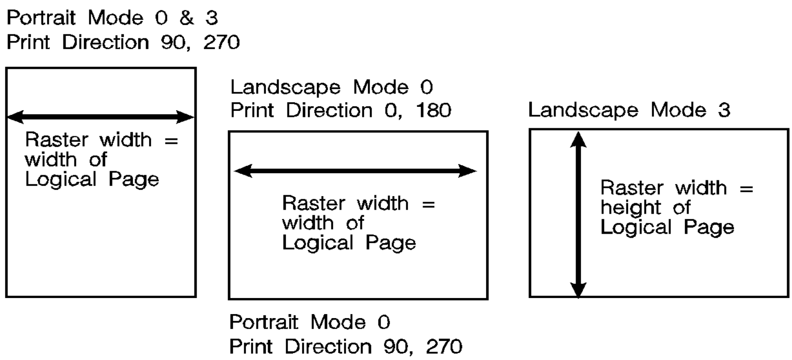


Figure 15-8 Maximum Raster Width

Start Raster Graphics Command

The Start Raster Graphics command identifies the beginning of the raster data and also specifies the left graphics margin.

$^{E}{}_{C}$ * r # A

\# =0- Start graphics at default left graphics margin X-position 0).

1 - Start graphics at current cursor position (current X-position).

Default = 0
Range = 0, 1 (values outside the range default to 0)

A value of 0 specifies that the left graphics margin is at the default left margin of the page (X-position 0). A value of 1 specifies that the left graphics margin is at the current X-position. In presentation mode 3, the location of the left graphics margin varies depending on the orientation.

Once a Start Raster Graphics command is received by the printer, raster graphics resolution, raster graphics presentation mode, raster height, raster width, and left raster graphics margin are fixed until an end raster graphics command is received.

Once in Raster Graphics Mode, PCL commands and text imply an End Raster Graphics ($^{E}{}_{C}$***rC**) except for the following commands:

- Transfer Raster Data
- Set Raster Compression Method
- Raster Y Offset

In addition, the following commands are ignored (i.e., locked out) while in Raster Graphics Mode and do not imply an End Raster Graphics command:

- Start Raster Graphics
- Set Raster Width
- Set Raster Height
- Set Raster Presentation Mode
- Set Raster Graphics Resolution

Notes

An implied End Raster Graphics resets the Raster Compression Method 3 seed row, but does not reset the Raster Compression Method nor the left raster graphics margin.

If source and/or transparency modes have been set, frequent start/end graphics commands in an image can result in a memory overflow condition.

Raster Y Offset Command

The Raster Y Offset command moves the cursor position vertically the specified number of raster lines from the current raster position in the raster area.

$^{E}{}_{C}$ * b # Y

=Number of raster lines of vertical movement

Default **=** N/A
Range **=** 0 - 32767

This command is recognized only while in raster graphics mode and only within the raster area.

Set Compression Method Command

The Set Compression Method command allows you to code raster data in one of four compressed formats: Run-length encoding, tagged imaged file format (TIFF) rev. 4.0, delta row compression, and adaptive compression. The choice of compression methods affects both the amount of code needed to generate a raster graphic image and the efficiency with which the image is printed.

$^{E}{}_{C}$& * b # M

=0- Unencoded
1 - Run-length encoding
2 - Tagged Imaged File Format (TIFF) rev. 4.0
3 - Delta row compression
4 - Reserved
5 - Adaptive compression

Default **=** 0
Range **=** 0 - 5 (values outside the range are ignored)

Unencoded (Method 0)

This is a simple binary transfer of data: no compression. Each bit describes a single dot. Bit 7 of the first byte corresponds to the first dot within the raster row, bit 0 corresponds to the eighth dot, and so on.

Note Compressed data formats allow for efficient transfer of data from the host system to the printer. However, compressed data formats do NOT reduce the amount of printer memory required to produce an image.

Run-length Encoding (Method 1)

Run-length encoding interprets raster data in pairs of bytes. The first byte of each pair is the repetition count for the data in the second byte. The second byte is the raster data to be printed.
A repetition count of 0 signifies the pattern in the data byte is not repeated (it occurs only once). A repetition count of 1 signifies the pattern occurs twice. The repetition count can range from 0 to 255 for a repetition of 1 to 256 times.

[(Repetition count byte 0-255)(pattern byte)] . [.] [ ]

Tagged Image File Format Encoding (Method 2)

Tagged image file format encoding interprets raster data as TIFF "Packbits." This format combines features of methods 0 and 1. A **control byte** precedes the raster data (pattern bytes). The control byte identifies whether the pattern byte(s) represent a byte that is to be repeated some number of times (up to 127), or represent some number of bytes (up to 127) which are to be printed as is (literal).

The sign of the number in the control byte identifies whether the byte or bytes that follow represent a literal pattern or byte to be repeated. A positive number (1 to 127) indicates that the bytes are literal. A negative number (-1 to -127), represented by the twos complement, indicates a repeated byte. The value of the number, if positive (literal), identifies the number of pattern bytes which follow the control byte; if negative (repeated), identifies the number of times to repeat the following byte. A pattern byte may be repeated up to 127 times; or up to 127 literal bytes may follow the control byte.

As mentioned, for a byte to be repeated, the control byte must be a negative value as represented by the twos complement. For example, to repeat a pattern three times would require the twos complement of the number 3. The twos complement is computed as follows. The binary of 3 is 00000011. Complement each bit to get 11111100, then add one to this value to produce 11111101, the twos complement. The decimal value of this number, 253, used in the control byte, produces a repetition of 3 bytes for a total of 4 occurrences of the pattern.

The range of numbers for the control byte is shown below.

Table 15-3 Literal Pattern Values

# of Bytes	Binary value	Decimal value
1	0000 0000	1
to	to	to
127	0111 1111	127

Table 15-4 No Operation Value

NOP value	Binary value	Decimal value
128 (-128)	1000 000	128

Table 15-5 Repeated Pattern Values

# of Repetitions	Binary value[1]	Decimal value
1 (-1)	1111 1111	255
to	to	to
127 (-127)	1000 0001	129

1. These negative values are represented by taking the twos complement of the value of the number.

Note Another method to calculate the number needed in the control byte for some number of repetitions is to subtract the number of desired repetitions from 256. For example, the control value for 3 repetitions (4 occurrences) of a byte is 256 minus 3 = 253.

A zero or positive value in the control byte means that the subsequent byte or bytes are non-replicated bytes of data. The value of the control byte *plus one* indicates the number of data bytes that follow. For example, a control byte of 0 means the following 1 byte is literal raster data. A control byte of 6 indicates that the following 7 bytes are literal raster data bytes.

TIFF encoding also allows you to include a non-operative (NOP) control byte, represented by the value –128. This byte is ignored, and the subsequent byte is treated as the new control byte.

Note It is more efficient to code two consecutive identical bytes as a repeated byte. If these bytes are preceded and followed by literal bytes, however, it is more efficient to code the entire group as literal bytes.

Examples: Run-length and TIFF Compression

The following examples show how a raster row can be coded using run-length and TIFF compression methods. Note that the compression examples use characters to represent the binary data stream.

Table 15-6

Byte Number	#1	#2	#3	#4	#5	#6	#7
Bits	01010101	01010101	01010101	01010101	01000001	01010100	01010100
ASCII	U	U	U	U	A	T	T

Unencoded

$^{E}{}_{C}$*r1A

$^{E}{}_{C}$*b0m7WUUUUATT

$^{E}{}_{C}$*rC

Run-length Encoding

$^{E}{}_{C}$*r1A

$^{E}{}_{C}$*b1m6W(3)U(0)A(1)T

$^{E}{}_{C}$*rC

TIFF Encoding

$^{E}_{C}$*r1A

$^{E}_{C}$*b2m6W(-3)U(0)A(-1)T or $^{E}_{C}$*b2m6W(-3)U(2)ATT

$^{E}_{C}$*rC

In the TIFF encoding example above, parenthetical expressions are used to identify control bytes. For example, the byte (-3) is shown to represent the control byte for a repetition (minus value) of 3. The actual value for this position is the decimal value 253. Additional “encoded” control bytes in this sequence include: (0) for decimal 0, (-1) for decimal 255, and (2) for decimal 2. The raster data (pattern) bytes are represented as by the ASCII character.

Delta Row Compression (Method 3)

Delta row compression identifies a section of bytes in a row that is different from the preceding row, and then transmits only that data that is different (the delta data). If a row is completely different from its preceding row, then the entire row must be sent as the delta (not very efficient); if only one bit is different, then only one byte is identified and sent. To reassemble the raster data rows, the printer takes the current row (referred to as the seed row) and makes the changes indicated by the delta data, to create the new row. The new row (which becomes the new seed row) is used by the next delta compression data to create another row.

A delta compression row consists of two parts, a command byte and the replacement bytes, as shown below:

[(Command byte)(1 to 8 Replacement bytes)]

The command byte identifies two things: 1) the number of replacement (delta) bytes that follow; and, 2) where to position the replacement byte string (the left offset). The replacement bytes are some number (up to eight bytes) of consecutive bytes that are used to create the new row from the seed row.

Table 15-7

Command Byte

7 5	4 0
Number of bytes to replace (1-8)	Relative offset from last untreated byte

If more than eight replacement (delta) bytes are needed, additional command byte/replacement bytes may be added, as shown below:

$^{E}{}_{C}$*3m#W [(Command Byte)(1 to 8 Replacement Bytes)][(Command Byte)(1 to 8 Replacement Bytes)]. . .

In the command byte, the upper three bits identify the number of replacement (delta) bytes (which can be 1 to 8 bytes). The lower five bits identify the location the replacement bytes are to be positioned. This position is identified as some number of bytes in, from the first untreated byte, referred to as the **offset**. For example, if there are 5 replacement bytes and the offset is 7, then the replacement bytes replace byte 7, 8, 9, 10, and 11 (the five bytes beginning at byte 7 from the seed row).

If there is more than one replacement in a row, the second offset is counted from the next untreated byte in the row: the first byte following the last replacement byte.

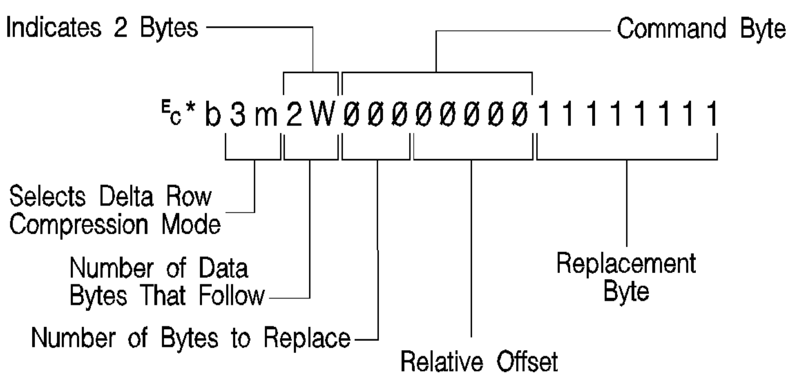


Figure 15-9

As mentioned, the offset is contained in the lower five bits of the command byte allows for offset values from 0 to 31. Compression mode allows offsets larger than 31 bytes as follows:

- An offset value of 0-30 indicates that the replacement bytes are offset from the 1st byte to the 31st byte.
- A value of 31 indicates that the next byte following the command byte is an additional offset byte which adds to the first (32) offset value. This allows offset values larger than 31. Also, if this second offset byte is set to 255 (all ones), additional offset bytes follow until the required offset value is obtained. When the formatter detects an offset byte less than 255, it is assumed to be the last offset value and the offset bytes are then totaled (added). The following example shows an offset larger than 31:

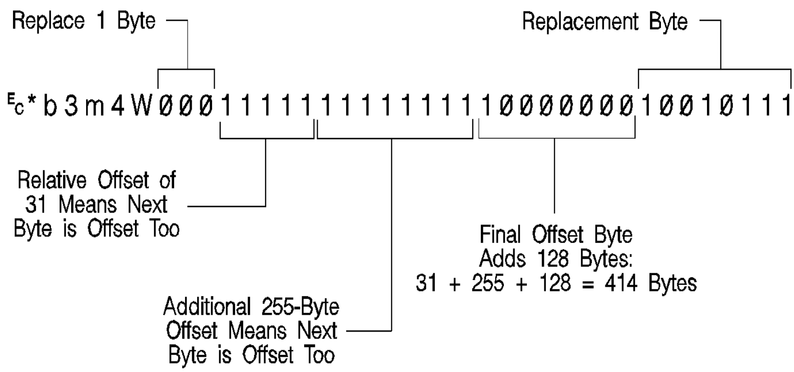


Figure 15-10

The total offset 414, which is the sum of the three offset values: 31 + 255 + 128.

Seed Row

The seed row is basically the current raster data row; the row being printed. It is maintained by the printer for use by delta row compression. The delta compression replacement bytes are applied to the seed row to create the new row. This new data row is printed and becomes the new seed row.

The seed row is updated by every raster graphic transfer, regardless of the compression method. This allows delta compression method to be mixed with other methods to achieve better compression performance.

Repeating a Row

$^{E}{}_{C}$*b0W

When using the delta compression method, it is possible to repeat or copy the previous raster row using the Raster Data Transfer command. This is accomplished by setting the Raster Data Transfer command, value field, to zero.

Printing A Zeroed Row (Setting the Seed Row to Zero)

$^{E}{}_{C}$*b1Y

It is possible to print a row of all zeros using the Raster Y-Offset command. Sending a Raster Y Offset command, with a value field of 1, sets the seed row to zero and prints the zeroed row. Note, that the next delta row is applied to a zeroed seed row.

Other cursor position moves set the seed row to zeros. (Remember, non-graphic cursor moves have the same effect as an end graphics command.)

Note

If the byte count of the Transfer Raster Data command value field is less than the number of bytes that can be replaced, the byte count has precedence. Also, if the last byte is a control byte, it is ignored. Therefore, **&esc*b1W** does not affect the seed row, but causes the previous row to be replicated.

Example: Delta Row Compression

The following example demonstrates how to compress the following data using the delta row compression. (The bytes highlighted in italic type indicate those bytes needing replacement — those bytes that are different from the previous row, the seed row.)

Table 15-8

Byte No.	0	1	2	3	4
Row 1	00000000	**11111111**	00000000	00000000	00000000
Row 2	00000000	11111111	**11110000**	00000000	00000000
Row 3	00001111	11111111	11110000	**10101010**	**10101010**

$^{E}{}_{C}$***r1A** – The *start raster graphics* command initializes the seed row to all zeros.

Row 1 — $^{E}{}_{C}$***b3m2W**(00000001)(11111111)

The **3m** selects the delta row compression method and the **2W** indicates that 2 bytes of data to follow. The first three bits of the first data byte, the command byte, signify a single byte replacement (all three bits are 0). The next five bits indicate an offset of 1 byte from the current position. The replacement byte follows and contains **11111111**.

Row 2 — $^{E}{}_{C}$***b2W**(00000010)(11110000)

The first three bits of the command byte indicate that one byte will be replaced, and the next five bits indicate a relative offset of 2, so the replacement will occur 2 bytes from the current position. The replacement byte follows and contains **11110000**.

Row 3 — $^{E}{}_{C}$***b5W**(00000000)(00001111)(00100010)(10101010)(10101010)

As in the other rows, the first three bits of the command byte are zero, indicating a single byte replacement. The five offset bytes indicate a relative offset of zero bytes. The replacement byte follows and is 00001111. The third byte is another command byte and the first three bits signify the replacement of two bytes (the top three bits are 001). The offset bits indicate an offset of two bytes from the current position. The fourth and fifth bytes are the two replacement bytes.

Adaptive Compression (Method 5)

Adaptive compression enables the combined use of any of the four previous compression methods (0 through 3), and it includes the ability to print empty (all zeros) rows or to duplicate rows.

Adaptive compression interprets a raster image as a block of raster data rather than as individual rows. The result of this interpretation is that the Transfer Raster Data ($^{E}{}_{C}$***b**#**W**) command is sent only once at the beginning of a raster data transfer, and the value field (**#**) identifies the number of bytes in the block (all rows). For the other compression methods, the Transfer Raster Data command is sent at the beginning of each row and the value field (**#**) identifies the number of bytes for that row only.

The size of a block is limited to 32,767 bytes. (32,767 bytes is the number of compressed bytes and not the size of the uncompressed data). To transfer greater than 32,767 bytes, send multiple blocks.

Adaptive compression uses three control bytes at the beginning of each row within the block. The first of these bytes, the command byte, identifies the type of compression for the row. The two following bytes identify the number of bytes or rows involved. The format for adaptive compression raster rows is shown below:

<command byte><# of bytes/rows - upper byte><# of bytes/rows - lower byte> ...
...<first raster row byte>...<last raster row byte>

The command byte designates the compression method, empty row, or row duplication. Command byte values are shown below.

Table 15-9

Value	Compression Operation
0 -	Unencoded
1 -	Run-Length Encoding
2 -	Tagged Image File Format (TIFF) rev 4.0
3 -	Delta row
4 -	Empty row
5 -	Duplicate row

For command byte values 0 - 3, the two **<# of bytes/rows>** bytes specify the number of bytes (row length) for the row. For command byte values 4 and 5, these bytes identify the number of empty or duplicate rows to print. The maximum value for these two bytes is 65,535; however, the image is clipped to the logical page. Thus, the value of these bytes should not exceed the maximum number of bytes/rows that can be printed on the current logical page size.

If an out of range command byte is encountered, the remainder of the block is skipped, the cursor is not updated, and the seed row is cleared.

Compression methods 0 - 3 are the compression methods used by the Set Compression Method command. Value fields 4 and 5 are features for the adaptive compression method and are explained below.

Empty Row

A command byte of 4, empty row, causes a row of zero's to be printed. The number of rows printed depends on the value contained in the two **<# of bytes/rows>** bytes following the command byte. The empty row operation resets the seed row to zero and updates the cursor position.

Duplicate Row

A command byte of 5, duplicate row, causes the previous row to be printed again. The row can be duplicated the number of times indicated by the value contained in the **<# of bytes/row>** byte. Duplicate Row updates the cursor position but does not change the seed row.

Adaptive Compression Operation Hints

Note Some HP LaserJet printers perform internal compression techniques to support full-page graphics. Refer to Chapter 1 of the *PCL 5 Comparison Guide* for specifics.

- The compression methods cannot be mixed within one raster row. A raster row must be compressed using only one method.
- The cursor position is updated with each row of the raster block. The cursor position is also incremented when a block count of less than 3 is sent.
- A Raster Y-Offset command moves the entire block of raster data and initializes the seed row to zeros. The seed row is set to zero even if the y-offset is zero.
- **B**lock size takes precedence over row length. If the row length of any line exceeds the block size, the row length is truncated to the block size.
- For duplicate and empty rows a row length value of zero does not update the cursor, however, the seed row is initialized to zero.
- If an unsupported command byte for a raster row is encountered, the remaining bytes for the block are skipped, the seed row is cleared, and the cursor is not incremented.
- For method 1, run length encoded, if the row length is odd, the cursor is incremented and the row data is skipped (thrown away), and the seed row is left unchanged.
- For method 1, a row length value of zero increments the cursor and zero fills the seed row.
- For method 2, TIFF, if row length terminates the data before the control byte value is satisfied (literal byte count greater than row length), the data following the control byte (if any) is printed as text. The cursor is incremented.
- For method 2 - If row length is equal to one, the one byte is consumed from the I/O and the cursor is incremented. The data is ignored and the seed row is zeroed.
- For method 3 - delta row compression, within an adaptive compression block, the seed row is updated by every raster compression method or type of row. For example, a row compressed with method 2, TIFF, updates the seed row, while the effect of an empty row initializes the seed row to zeros. Maintaining the seed row allows method 3 to be mixed with other methods to achieve optimal compression performance.

- For method 3 - Since delta row compression requires that the seed row be available whenever raster graphics mode is entered, the seed row is initialized to zeros upon raster graphics mode entry ($^{E}{}_{C}$*r#A). The seed row is also initialized upon receipt and completion of each raster block.
- For method 3 - If the row length terminates the data before the control byte value is satisfied (literal byte count greater than row length), the data following the control byte (if any) is printed as text. The cursor is incremented.
- For method 3 - if the row length is equal to one, the current row is duplicated, and the cursor is incremented.

Transfer Raster Data Command

The Transfer Raster Data command is used to transfer a row of raster data to the printer.

E_C * b # W   [*raster data*]

Default = N/A
Range = 0 - 32767

The value field (#) identifies the number of bytes in the raster row. These bytes are interpreted as one row of raster graphics data that is printed at the current Y position at the left raster graphics margin. Upon completion of this command, the cursor position is at the beginning of the next raster row at the left raster graphics margin.

Within the raster data, each bit describes a single dot. The most significant bit (bit 7 is the most significant, bit 0 is the least significant) of the first byte of data corresponds to the first dot within the row. If a bit is set to 1, the corresponding dot is printed. Each dot of the raster data is expanded according to the specified raster resolution.

Raster graphics is independent of the text area and perforation skip mode – these boundaries are ignored.

Raster graphic images, raster height, and raster width are limited to the printable area; images that extend beyond the printable area are clipped.

Note The byte count of the value field in the Transfer Raster Data command has precedence over the literal or the command byte, byte count. For example, the command,

E_C*b2m3W [*binary data*]

sets compression method=2 and sends 3 bytes of raster data for the row. Suppose the binary data appears as follows:

00000010 00000001 00000001 00000001

The control (first) byte value of +2 indicates that 3 bytes of literal (unencoded) raster data will follow. The Transfer Raster Data command, however, specified only three bytes total (including the control byte) in the raster row. The control byte and the following two data bytes are read, and the remaining data byte is ignored.

Notes If the last byte indicated by the value field in the Transfer Raster Data command is a control byte, that byte is ignored.

If a Transfer Raster Data command is received without an accompanying Start Raster Graphics command, any preceding start raster values are used (such as left graphics margin, raster height and width, etc.).

End Raster Graphics Command

The **End Raster Graphics** command signifies the end of a raster graphic data transfer.

$^{E}{}_{C}$ * r C

- Receipt of this command causes 5 operations:
- Resets the raster compression seed row to zeros.
- Moves the cursor to the raster row immediately following the end of the raster area (if a source raster height was specified).
- Allows raster commands which were previously locked out to be processed.
- Sets compression mode to 0 (no compression)
- Defaults the left graphics margin to X-position 0.

Notes This command is a modified version of the $^{E}{}_{C}$***rB End Raster Graphic**s command. This new version ($^{E}{}_{C}$*rC) performs two additional operations: 1) it resets the compression mode to 0, and 2), it defaults the left graphics margin to 0.

This command ($^{E}{}_{C}$*rC) is not supported by the HP LaserJet III or the HP LaserJet IIID printers. Use the $^{E}{}_{C}$***rB End Raster Graphics** command to terminate raster graphic data transfers for these printers.

Refer to the “PCL Feature Support Matrix” in Chapter 1 of the *PCL 5 Comparison Guide* for specific printers which support these commands.

Raster Graphics Example

To transfer an unencoded raster graphic image (see Figure 15-11) in the shape of an arrow, perform the following steps:

Table 15-10

1. Position the cursor:

EC*p300x400Y	This moves the cursor to PCL Unit position (300, 400) within the PCL coordinate system.

2. Specify the raster graphics resolution:

EC*t75R	This sets the raster graphics resolution to 75 dots-per-inch.

3. Specify the raster graphics presentation method:

EC*r0F	This specifies that the raster graphics is printed in the orientation of the logical page.

4. Specify the left raster graphics margin:

EC*r1A	This sets the left graphics margin to the current X position (300).

5. Transfer the raster data to the printer:

Divide the image into dot rows and transfer each dot row to the printer as a string of bytes, as illustrated on the following page.

6. Signify the end of the raster graphic image transfer:

EC*rC	This example prints the arrow as shown in Figure 15-11.

Table 15-11 Example of Raster Graphic Image Data

	Raster Image Data				Command Data
Dot Row	**byte 1**	**byte 2**	**byte 3**	**byte 4**	**Decimal Equivalent**
1	00000000	00000000	10000000	00000000	EC*b4W[0, 0,128, 0]
2	00000000	00000000	11000000	00000000	EC*b4W[0, 0,192, 0]
3	00000000	00000000	11100000	00000000	EC*b4W[0, 0,224, 0]
4	00000000	00000000	11110000	00000000	EC*b4W[0, 0,240, 0]
5	00000000	00000000	11111000	00000000	EC*b4W[0, 0,248, 0]

Table 15-11 Example of Raster Graphic Image Data (continued)

6	00000000	00000000	11111100	00000000	EC*b4W[0, 0,252, 0]
7	00000000	00000000	11111110	00000000	EC*b4W[0, 0,254, 0]
8	00000000	00000000	11111111	00000000	EC*b4W[0, 0,255, 0]
9	00000000	00000000	11111111	10000000	EC*b4W[0, 0,255,128]
10	11111111	11111111	11111111	11000000	EC*b4W[255,255,255,192]
11	11111111	11111111	11111111	11100000	EC*b4W[255,255,255,224]
12	11111111	11111111	11111111	11110000	EC*b4W[255,255,255,240]
13	11111111	11111111	11111111	11111000	EC*b4W[255,255,255,248]
14	11111111	11111111	11111111	11111100	EC*b4W[255,255,255,252]
15	11111111	11111111	11111111	11111110	EC*b4W[255,255,255,254]
16	11111111	11111111	11111111	11111111	EC*b4W[255,255,255,255]
17	11111111	11111111	11111111	11111111	EC*b4W[255,255,255,255]
18	11111111	11111111	11111111	11111110	EC*b4W[255,255,255,254]
19	11111111	11111111	11111111	11111100	EC*b4W[255,255,255,252]
20	11111111	11111111	11111111	11111000	EC*b4W[255,255,255,248]
21	11111111	11111111	11111111	11110000	EC*b4W[255,255,255,240]
22	11111111	11111111	11111111	11100000	EC*b4W[255,255,255,224]
23	11111111	11111111	11111111	11000000	EC*b4W[255,255,255,192]
24	00000000	00000000	11111111	10000000	EC*b4W[0, 0,255,128]
25	00000000	00000000	11111111	00000000	EC*b4W[0, 0,255, 0]
26	00000000	00000000	11111110	00000000	EC*b4W[0, 0,254, 0]
27	00000000	00000000	11111100	00000000	EC*b4W[0, 0,252, 0]
28	00000000	00000000	11111000	00000000	EC*b4W[0, 0,248, 0]
29	00000000	00000000	11110000	00000000	EC*b4W[0, 0,240, 0]
30	00000000	00000000	11100000	00000000	EC*b4W[0, 0,224, 0]
31	00000000	00000000	11000000	00000000	EC*b4W[0, 0,192, 0]
32	00000000	00000000	10000000	00000000	EC*b4W[0, 0,128, 0]

The brackets and commas are not part of the raster data command; they are used only to delineate the data.

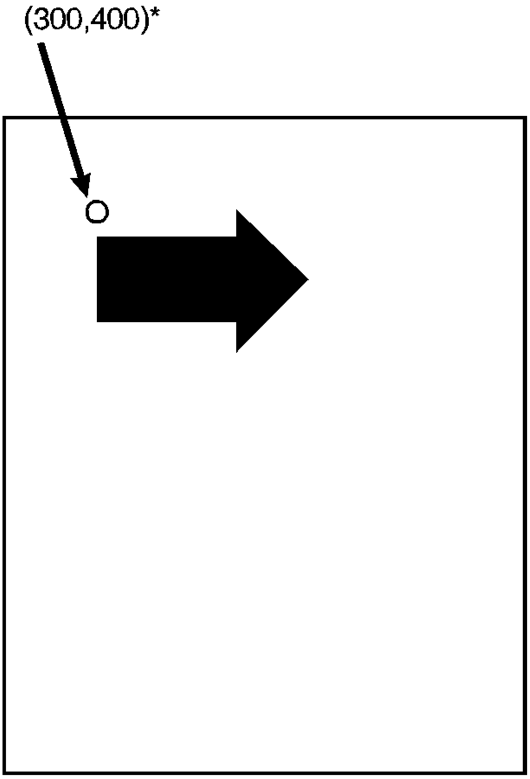


Figure 15-11 Example of Raster Graphic Image Data

16 Status Readback

Introduction

This chapter describes the PCL status readback features. PCL status is requested from the printer with the commands described in this chapter. Following a status request, the printer generates a status response. This response consists of ASCII data which is sent directly from the printer, through the I/O, back to the host. Status readback allows you to obtain information from the printer such as: available printer (user) memory, current available fonts and symbol sets, and the ID numbers of downloaded macros and user-defined patterns to verify their presence.

The contents of this chapter are listed below:

- Flush All Pages Command
- Echo Command
- Status Readback Programming Hints

For status readback to work, the host system must contain a bi-directional driver to receive status response data. In network operations, some printing environments do not support bi-directional communication, such as many printer sharing devices which spool data, and some network operating systems. Applications designed to incorporate PCL status readback should be designed to function correctly in situations where no response is possible.

Note For the stand-alone DOS personal computer, bi-directional driver/applications are required to access the parallel/serial I/O communication data for not only sending to the printer but also receiving data (status) back from the printer.

FTo obtain status information you must initiate a request by sending either a request for memory status or a request for an entity status, as described on the following pages.

Memory Status Request

It is possible to identify the amount of available user memory using the memory status request. Being able to identify the available memory enables a user to determine whether sufficient memory is available for the entity being downloaded, potentially avoiding a printer memory overflow condition (control panel error 20, memory overflow).

An example memory request with its associated response is shown below.

Table 16-1 Memory Status Request Example

Description	I/O Data
Memory request (Free Space command) sent to printer from host	$^{E}{}_{C}$*s1M

Table 16-1 Memory Status Request Example (continued)

Status response sent from printer to host	PCL INFO MEMORY TOTAL=100000 LARGEST=25000

Entity Status

It is possible to request status for the printer's entities. An **entity** is a font, symbol set, macro, or user-defined pattern stored in the printer. Each individual entity request is limited to one specific entity, and is further limited to a specific location. To request entity status, you must send the entity status readback commands to identify a **location type** and a **location unit**, and then send the entity request command.

Location type refers to the memory locations which store entities. These memory locations include internal ROM, RAM (for downloaded entities), cartridges, user-installable ROMs (SIMMs), and one additional location identified as "currently selected.“ Currently selected identifies the entity which is active, such as the font or user-defined pattern last selected. (Currently selected does not apply to macros or symbol sets).

Location unit refers to a specific location (or device) within the location type. For example, location unit "1" for location type "cartridge,“ identifies the left cartridge on a printer with two cartridges; or, unit "1" for location type "downloaded," identifies the temporary fonts (as opposed to permanent).

The location type and unit are described in detail under the location type and unit status readback command descriptions provided later in this chapter.

Once the location type and unit are specified, the status can be requested using the Inquire Status Readback Entity command. This command identifies the entity (font, symbol set, macro, or user-defined pattern) and causes the printer to send the response.

A basic entity status request is shown in the example below. This example identifies a status request for downloaded permanent fonts. The example also includes a status readback response at the bottom. Note that this example shows only one possible request/response; for more detailed information, refer to the status readback command descriptions provided later in this chapter.

Table 16-2 Entity Status Request Example

Operation	**Example PCL Command**	**Comments**
Set location type	$^{E}{}_{C}$*s4T	This Set Location Type command sets the location type to “downloaded.”
Set location unit	$^{E}{}_{C}$*s2U	This Set Location Unit command sets the location unit to “permanent.”
Identify entity which initiates status readback (Inquire Entity)	$^{E}{}_{C}$*s0I	The Inquire Status Readback Entity command selects status for “fonts” and causes the printer to buffer the response.

Table 16-3

Printer status res-ponse	PCL INFO FONTS SELECT=“<Esc>(8U<Esc>(s1p__v0s0b4120T <Esc>(7X” SELECT=“<Esc>(10U<Esc> __v0s0b4157T<Esc>(21X”	Two permanent downloaded fonts exist in the printer: University Roman (4120T) and Dom Casual (4157T).

In addition to the normal status responses, if an invalid request is made, the printer returns an error response. Error responses are described under the Inquire Status Readback Entity command, later in this chapter.

Status Response

When the printer receives a status request (command), it processes that request and forms the response data. This data is then stored in an I/O status buffer. The response is saved in this buffer until it is either read (by any user) or the printer is turned off.

In addition to clearing the status buffer by reading the status response or by turning off the printer, status responses are cleared if one the following settings are changed:

- Printer resolution (600/300)
- Page protection
- Language personality

Status requests and their associated response are processed in the order in which they are received.

The number of responses the printer can buffer varies, depending on internal printer operations. When requesting status, especially in the case where multiple applications or users are sharing one printer, HP recommends that you request one item, and then read its response prior to making another request.

Note Since the printer may contain a status response requested by a previous application's operation, it is important to use the Echo command (described later in this chapter) to synchronize your application's request with the printer's responses. Refer to the Echo command description and to the "Programming Hints" section at the back of this chapter for additional information.

Note The number of status responses a printer can buffer varies from printer to printer. The HP LaserJet 4 printer can store 5 responses. Refer to the *PCL 5 Comparison Guide* for printer specific information.

A status response is returned to the printer I/O port from which it was requested.

Status Response Syntax

All status responses start with the letters "PCL" followed by a Carriage Return control code (decimal 13; <CR>) and a Line Feed control code (decimal 10; <LF>). In addition, each remaining line of the response is terminated by a Carriage Return and Line Feed control code. Finally, status responses are terminated by a Form Feed control code (decimal 12; <FF>). The basic syntax for the status response is shown below.

```
PCL<CR><LF>
INFO TITLE<CR><LF>
KEYWORDn=DATAn<CR><LF>
KEYWORDn=DATAn<CR><LF>
   ⋮
<FF>
```

TITLE, *KEYWORDn*, and *DATAn* are strings that vary depending on the particular status readback command being executed. Each status readback response has one or more keyword lines associated with it. A slight variation of this is the Echo command response. This response is shown below.

```
PCL<CR><LF>
ECHO ValueField<CR><LF>
<FF>
```

ValueField is the decimal number taken from the Echo command value field.

Note Keywords may be added for future printers that are not documented here. Applications that use status readback should be designed to ignore lines with keywords they do not recognize.

Two example status responses are shown below.

```
PCL<CR><LF>
INFO MEMORY<CR><LF>
TOTAL=100000<CR><LF>
LARGEST=25000<CR><LF>
<FF>
```

```
PCL<CR><LF>
INFO FONTS<CR><LF>
SELECT="<Esc>(8U<Esc>(s0p10.00h12.00v0s0b3T"<CR><LF>
SELECT="<Esc>(0N<Esc>(s0p16.67h8.5v0s0b0T"<CR><LF>
SELECT="<Esc>(s1p__v1s0b4101T<Esc>(78X"<CR><LF>
SYMBOLSETS="0D,0I,0N,0S,0U,1E,1F,1G,..."<CR><LF>
SELECT="<Esc>(8U<Esc>(s1p__v0s3b4148T"<CR><LF>
<FF>
```

Notes

In the example listings, "<CR>," "<LF>," and "<FF>" identify the Carriage Return (decimal 13), Line Feed (decimal 10), and Form Feed (decimal 12) control codes. The "<Esc>" following the "SELECT=" keyword is a five character representation (the printable characters: "<," "E," "s," "c," and ">") and does not identify an escape control code (decimal 27).

The examples in the remainder of this chapter do not show the status response line termination ("<CR><LF>") control codes, or the termination ("<FF>") control code.

Example responses in this chapter for "SYMBOLSETS=" do not list the complete list of internal available symbol sets, only a partial list: "0D,0I,0N,0S,0U,1E,1F,1G,..." The internal symbol sets are printer dependent and may vary from printer to printer. Refer to the *PCL 5 Comparison Guide* for printer specific information.

Set Status Readback Location Type Command

The Set Location Type command sets the status location type to the specified value. Location type is used in conjunction with the location unit to identify an entity location for a status request (Inquire Status Readback Entity command).

$^{E}{}_{C}$ * s # T

\# =0- Invalid location
1 - Currently selected
2 - All Locations
3 - Internal
4 - Downloaded Entity
5 - Cartridge
7 - User-installable ROM device (SIMMs)

Default **=** 0
Range **=** 0 - 5, 7

If a value outside the range is received, the location type is set to 0.

When the location type is 0 and an Inquire Entity command is received, an error response is generated (refer to the “Status Response Error Codes” section later in this section for additional information).

A printer reset returns the location type setting to 0.

The printer retains the location type setting. If the Set Status Readback Location Type command is not sent to change the setting for an entity request, then the existing location type setting is used.

Set Status Readback Location Unit Command

The Set Location Unit command sets the status location unit to the specified value. Location unit is used in conjunction with the location type to identify an entity location for a status request (Inquire Status Readback Entity command). Note that the unit value is interpreted differently, depending on the location type specified.

$^{E}{}_{C}$ * s # U

Table 16-4 Set Status Readback Location Unit Command

Location Type		Location Unit		
0	#	=	*	Invalid location
1		=	*	Currently selected
2		=	*	All Locations
3		=	0	All internal
4		=	0	All downloaded
		=	1	Temporary downloaded
		=	2	Permanent downloaded
5		=	0	All cartridges
		=	1	Highest priority cartridge
			⋮	⋮
			n	Lowest priority cartridge
7		=	0	All SIMMs
		=	1	Highest priority SIMM
			⋮	⋮
			n	Lowest priority SIMM

Default **=** 0
Range **=** 0 through n, where n is printer dependent. Refer to the *PCL 5 Comparison Guide*.

* - For location type values 0, 1 and 2, the unit value is ignored; it may be any value.

A value of 0 indicates all units of the location type.

A printer reset ($^{E}{}_{C}$E) returns the location unit to its default value, 0.

The printer retains the location unit setting. If this command is not sent to change the setting for an entity request, then the existing location unit setting is used.

Note The location type and unit may be set in any order. Invalid combinations are not determined until the Inquire Entity command is received. Therefore, even if the unit value is out of range, the unit is set to that value so that an appropriate error response is sent when the Inquire Entity command is received.

Inquire Status Readback Entity Command

The Inquire Entity command identifies the entity type and causes the printer to create a status response for the entity specified in the status readback location (type and unit).

$^{E}{}_{C}$ * s # I

\# =0- Font
1 - Macro
2 - User-defined pattern
3 - Symbol Set (for unbound scalable fonts)
4 - Font Extended

Default **=** NA
Range **=** 0 - 4

The entity status responses for the value field parameters vary depending on the setting of the location type and location unit (refer to the Set Location Type and Set Location Unit commands described on the preceding pages).

The entity status and error responses are described on the following pages.

Entity Status Responses

The status response for an Inquire Status Readback Entity command varies depending on the type of entity requested. The status responses for font, font extended, macro, user-defined pattern, and symbol set are described below.

Font Response

The status response information returned for font (inquire entity value field 0) varies. Depending on whether the printer's font is a bitmap, unbound scalable, or bound scalable, different keywords are returned. The response is also somewhat different if the location type is set to 1 (currently selected, as explained in more detail later). The list of possible keywords for a font request is shown below.

SELECT=
SYMBOLSETS=
LOCTYPE=
LOCUNIT=

An example of an inquire entity font (entity type 0) status response is shown below. Notice the variation in the keyword lines for the three fonts, listed in order below:

Internal, bitmap, Line Printer
Internal, unbound scalable, CG Times italic
Downloaded, bound scalable, (CG Palacio)

Font status:

```
PCL
INFO FONTS
SELECT="<Esc>(8U<Esc>(s0p16.67h8.5v0s0b0T"
SELECT="<Esc>(s1p__v1s0b4101T"
SYMBOLSETS="0D,0I,0N,0S,0U,1E,1F,1G,..."
SELECT="<Esc>(1U<Esc>(s1p__v0s0b4111T<Esc>(21X"
  ⋮
```

In a font status response, individual fonts are identified by the "SELECT=" keyword line, as shown in the example above. This line identifies the font by specifying the font selection characteristics (symbol set, spacing, pitch, height, style, stroke weight, and typeface). The characteristics are listed as they would be sent to the printer to select the font (in priority order, highest priority to lowest). "SELECT=" is returned for all font types.

As mentioned, variations in the font response occur as a result of the font being a bitmap, bound scalable, unbound scalable, soft (downloaded) font, or the location type set to 1 (currently selected). These variations include slight differences in the information included in the “SELECT=” line, with the addition of different keywords (“SYMBOLSETS=” “LOCTYPE=” and “LOCUNIT=”). These variations are described in detail in the following paragraphs.

Notes

In the status response, an escape character is represented by the five characters: “**&**”.

Keywords which follow a “SELECT=” keyword apply to the font identified by that “SELECT=” keyword.

Bitmap Fonts

For bitmap fonts, the “SELECT=” line is returned (which identifies the font selection characteristics as described above). For example, the printer’s internal, bitmap, Roman-8 Line Printer font would be returned as shown below.

```
PCL
INFO FONTS
SELECT=“<Esc>(8U<Esc>(s0p16.67h8.5v0s0b0T”
```

Also see soft font description below.

Scalable Fonts

For bound scalable fonts, two underscores, “...s 1 p __ v 0 s...,” in the “SELECT=” line indicate that the font is a bound scalable. In addition, only the relevant characteristics are listed. Depending on whether the font is proportional or fixed spaced, only either height or pitch is listed. For example, the printer’s internal scalable Symbol font (proportionally spaced) would be returned as shown below.

```
PCL
INFO FONTS
SELECT=“<Esc>(19M<Esc>(s1p__v0s0b16686T”
```

Also see the soft font description below.

Unbound Scalable Fonts

For unbound fonts, since multiple symbol sets can be associated with an unbound font, the “SYMBOLSETS=” keyword is added to list the available symbol sets and the symbol set sequence is no longer present in the “SELECT=” line. For example, the printer’s internal unbound scalable Courier (medium, fixed spaced) font and CG Times (bold italic, proportional spaced) fonts would be returned as shown below.

```
PCL
INFO FONTS
SELECT=“<Esc>(s0p__h0s0b4099T”
SYMBOLSETS=“0D,0I,0N,0S,0U,1E,1F,1G,...
SELECT=“<Esc>(s1p__v1s3b4101T”
SYMBOLSETS=“0D,0I,0N,0S,0U,1E,1F,1G,...”
```

Soft Fonts

For soft fonts (bitmap, bound scalable or unbound scalable), the font ID number is also included at the end (a font downloaded with an ID of 27 would be presented as “ ...<Esc>(27X ”). For example, a downloaded bitmap Courier with a font ID of 39 and a downloaded, unbound scalable, Dom Casual typeface with a font ID of 78 would be returned as shown below.

```
PCL
INFO FONTS
SELECT=“<Esc>(8U<Esc>(s0p10.00h12.0v0s0b3T<Esc>(39X”
SELECT=“<Esc>(s1p__v1s0b4148T<Esc>(78X”
SYMBOLSETS=“0D,0I,0N,0S,0U,1E,1F,1G,...”
```

Location Type 1 (Currently Selected) Font

For a status location type 1 (currently selected) font, only a single font, the printer’s currently selected font, is returned. The keywords and data returned are described as follows.

- “LOCTYPE=” and “LOCUNIT=” are returned after the “SELECT=” line for each font.
- “SELECT=” line changes for bound and unbound scalable fonts.
  - For all scalable fonts, the actual size (either height or pitch) is listed in place of the underscores.
  - For unbound scalable fonts, the symbol set currently bound to the font is listed.

- For all fonts, if the font is a secondary font, then the “(“ characters are replaced by ”)” characters in the “SELECT=” line.

“LOCTYPE=” identifies the location type of the currently selected font. The value returned corresponds to the value field of the Set Status Readback Type command.

“LOCUNIT=” identifies the location unit of the currently selected font. The value returned corresponds to the value field of the Set Status Readback Unit command.

For example, if the printer’s currently selected font is a bold 14 point Presentation bitmap font selected from a cartridge, and a font entity request with the location type set to 1 is made, the response would be returned as shown below.

```
PCL
INFO FONTS
SELECT=“<Esc>(1U<Esc>(s0p10.00h14.0v0s3b11T
LOCTYPE=5
LOCUNIT=1
```

For example, assume the printer’s currently selected font is unbound scalable Dom Casual font with the following characteristics:

- downloaded (font ID of 78)
- temporary
- Roman-8 (specified for printing)
- secondary font
- 18 point

If an entity request is made with the location type set to 1 (currently selected), the following response is returned.

```
PCL
INFO FONTS
SELECT=“<Esc>)8U<Esc>)s1p18.00v1s0b4148T<Esc>)78X”
LOCTYPE=4
LOCUNIT=1
```

Font Extended Response

The font extended response (inquire entity value field 4) provides a way to return the name and internal ID number of the font, as well as the "SELECT=" line. It is the same as a font status response (inquire entity value field 1), in that "SELECT=," "SYMBOLSETS=," "LOCTYPE=," and "LOCUNIT=" are returned in the same manner. They are not re-described here; refer to the "Font Response" section for a description of their operation. However, for a font extended request, two additional keywords, "DEFID=" and "NAME=" are returned as described below.

"**DEFID=**" identifies the font's internal ID number. This is the number which appears on the font printout. It is the number used to select the font as the default font from either the printer's control panel or from PJL (refer to the *Printer Job Language Technical Reference Manual* for information on PJL font selection). The "DEFID=" number consists of two parts, a location and an ID number, such as "I 21," where "I" is the location and "21" is the font's internal ID number. The possible locations are listed below:

Table 16-5

I	-	Internal
C	-	Cartridge - single cartridge printers
Cn	-	Cartridge - multiple cartridge printers where n is printer specific
S	-	Permanent soft fonts
Mn	-	SIMMs where n is the number of the SIMM slot for example: M2 is SIMM in #2 slot.
NONE	-	Temporary soft fonts

Note

The cartridge (Cn) and SIMMs (Mn) location identifiers are printer specific. Refer to Chapter 1 of the *PCL 5 Comparison Guide* for printer-specific values for "n."

Since temporary soft fonts do not have an internal ID number (they cannot be selected as the default from the control panel or PJL), NONE is returned as their "DEFID=NONE".

The font's internal ID number is assigned to fonts by the printer. This number is different than the soft font download ID number assigned to a downloaded font using the Font ID (^{E}C*c#D) command.

"**NAME=**" is returned for font extended status requests only. It identifies the name of the font, such as, Courier, Times Roman, Univers, etc., and its treatment (such as BdIt - bold italic) as listed in the font printout.

For example, the font extended response for the printer's internal bitmap Line Printer font and a downloaded unbound CG Palacio temporary font would be returned as shown below.

```
PCL
INFO FONTS EXTENDED
SELECT="<Esc>(8U<Esc>(s0p10.00h12.0v0s0b0T"
DEFID="I 45"
NAME="Line Printer"
SELECT="<Esc>(s1p__v1s3b4111T<Esc>(7X"
SYMBOLSETS="0D,0I,0N,0S,0U,1E,1F,1G,..."
DEFID=NONE
NAME="CG Palacio   BdIt"
SELECT="...
⋮
```

Or, for example, a font extended response with the location type set to 1 (currently selected) where the currently selected font is the internal CG Times (18 point specified for printing) bold, the response would be returned as shown below.

```
PCL
INFO FONTS EXTENDED
SELECT="<Esc>(8U<Esc>(s0p18.00h0s3b4101T"
DEFID="I 002"
NAME="CG Times   Bd"
LOCTYPE=3
LOCUNIT=1
```

Macro Response

The status response for macros (inquire entity value field 1) lists all of the macro IDs ("IDLIST=") for the macros in the specified location.

Note Status location type 1 (currently selected) is an invalid location for macros and returns an error ("ERROR=NONE").

A macro status response might appear as shown below.

```
PCL
INFO MACROS
IDLIST="1,3,8,29,32"
```

User-Defined Pattern Response

The status response for user-defined patterns (inquire entity value field 2) lists all of the user-defined pattern IDs ("IDLIST=") for the patterns in the specified location.

A user-defined pattern response might appear as shown below.

```
PCL
INFO PATTERNS
IDLIST="1,2,9,13,27,456"
```

If the location type is set to 1 (currently selected), then "LOCTYPE=" and "LOCUNIT=" lines are added.

"LOCTYPE=" is returned for a status location type 1 request only. It identifies the location type of the currently selected pattern.

"LOCUNIT=" is returned for a status location type 1 request only. It identifies the location unit of the currently selected pattern.

A user-defined pattern response for the currently selected pattern might appear as shown below.

```
PCL
INFO PATTERNS
IDLIST="88"
LOCTYPE=4
LOCUNIT=2
```

If the current pattern is set to one of the internal HP-defined patterns (no pattern ID number assigned), then no number is available and the response, "ERROR=NONE" is returned.

```
PCL
INFO PATTERNS
ERROR=NONE
```

Symbol Set Response

The response for symbol sets (inquire entity value field 3) lists all of the symbol set IDs (“IDLIST=”) for all of the symbol sets that can be bound to unbound scalable fonts in the specified location (type and unit).

A symbol set response might appear as shown below.

```
PCL
INFO SYMBOLSETS
IDLIST=“0U,2K,8M,8U,11U”
```

Notes

Status location type 1 (currently selected) is an invalid location for unbound font symbol sets and returns an error (“ERROR=NONE”).

Example responses in this chapter for “SYMBOLSETS=” do not list the complete list of internal available symbol sets, only a partial list: “0D,0I,0N,0S,0U,1E,1F,1G,...” The internal symbol sets are printer dependent and may vary from printer to printer; refer to the *PCL 5 Comparison Guide* for printer specific information.

Entity Error Codes

If you request out-of-range values in the command value fields, or if the entity is unsupported or does not exist, or if the request is inappropriate, the printer responds with one of four possible errors:

- **Invalid Entity**
- **Invalid Location**
- **None**
- **Internal Error**

ERROR=INVALID ENTITY

If the entity type specified in the escape sequence is out of range or unsupported, an invalid entity error is returned. For example, if the Inquire Entity command contained an out of range value of 8 ($^{E}{}_{C}$***s8I**), the following error response would be generated.

```
PCL
INFO ENTITY
ERROR=INVALID ENTITY
```

ERROR=INVALID LOCATION

If the entity type is valid but the location (either the type, the unit, or the combination) is invalid or if the specified device is not installed, an invalid location error is returned. For example, if you requested a status for a cartridge type but identified an out of range location unit of 9 ($^{E}{}_{C}$***s5t9U**), the following error response would be generated:

```
PCL
INFO FONTS
ERROR=INVALID LOCATION
```

ERROR=NONE

If the entity type and location are valid, but there are no entities of the specified type in that location, or if the type is inappropriate for the specified entity (internal user-defined pattern or currently selected macro), then an error response is generated. For example, if you request the downloaded symbol sets and there are no downloaded symbol sets, the following error response would be generated:

```
PCL
INFO SYMBOLSETS
ERROR=NONE
```

ERROR=INTERNAL ERROR

The status response for some requests can be fairly large (such as for fonts). In processing status responses, if the printer runs out of memory, an internal error is returned, as shown below:

```
PCL
INFO SYMBOLSETS
ERROR=INTERNAL ERROR
```

Note The error conditions described above are the only conditions for which an error response is generated. If you make a syntax error in the escape sequence, or send a command which the printer cannot interpret, the printer ignores the command and no error response is given.

Free Space Command

The Free Space command returns the amount of available memory. This response returns two values: the total available memory, and the largest available block of memory (refer to the “Memory Status Response” section later in this chapter for additional information).

$^{E}{}_{C}$ * s 1 M

If a value other than 1 is sent, this command returns an error (“ERROR = INVALID UNIT,” refer to “Memory Error Response” section later in this chapter for additional information).

To identify whether the printer has enough memory available for a job, you can send the Free Space command to compare the space available with that needed. You can also identify how much memory an entity or any other data uses by checking the amount of free memory prior to downloading the data, then downloading the data, and checking memory again. The difference in these values represents the approximate memory needed.

Note Many conditions can cause the available memory to change or appear different. Some of these conditions are listed below.

The actual printer memory required to store an item varies slightly based on printer memory fragmentation and other internal printer conditions.

Different printer models use different methods to store data. Thus, the amount of memory required to store the same amount of data may be slightly different in different printers.

While the printer is processing page data, the available memory is constantly changing due to the printer receiving new data, processing existing data, and adding new characters to the font cache, etc. Under these conditions, the available memory may change by the time the memory response is returned.

If a PostScript SIMM is installed in the printer, some memory is not reported for a Free Space command response. This memory is not reported as part of the free memory for a PCL status readback response, however, this memory is available for PCL use. Thus, it is possible for all or part of the downloaded data to be stored in this section of unreported memory and not change the memory response size.

If you determine there is insufficient memory to hold the data to be downloaded, some action is required. One method to make more memory available is to send the Flush All Pages command. This causes the printer to clear (process) the current page data from memory without accepting any new data for processing (refer to the “Flush All Pages Command” described later in this section). Another, more comprehensive method to clear memory is to send the Printer Reset ($^{E}{}_{C}$E) command. This not only removes data (deletes all temporary entities and the font cache) but also restores the User Default Environment settings (refer to the Printer Reset command in Chapter 4, “Job Control Commands”).

Note To print characters from a scalable font, the printer converts the scalable character outlines into sized bitmaps. These bitmapped characters are created on a character-by-character basis as they are needed for printing and are stored in memory. As more pages are printed using more fonts, the bitmaps consume more memory. The bitmap characters used on the first page of a job can remain in memory until the end of a job. These stored bitmap characters are referred to as the font cache.

Note When an HP LaserJet 4 printer (or later) reaches a memory low condition it automatically deletes all of the font cache. It is possible to delete the cache immediately using a Printer Reset command. A Printer Reset clears the font cache, clears temporary entities, and restores the user default environment.

Memory Status Response

A Free Space status response returns two values:

```
TOTAL=
LARGEST=
```

“TOTAL=” identifies the total available user memory (in bytes). This value includes the largest block available and all smaller blocks.

Note Data downloaded to the printer is stored in a block (continuous section of free memory). If the printer does not have a large enough block to store the data, then the data is discarded and a memory error results.

“LARGEST=” identifies the largest continuous block of available memory (in bytes).

An example response is shown below:

```
PCL
INFO MEMORY
TOTAL=100000
LARGEST=25000
```

The above example indicates that the printer has 100,000 bytes of available memory and the largest continuous block is 25,000 bytes.

Memory Error Response

If the Free Memory command value field is out of range (not 1), then the memory status response returns an invalid unit error. For example, if the Free Space command with a value of 2 ($^{E}{}_{C}$***s2M**) were sent, the following error would be returned:

```
PCL
INFO MEMORY
ERROR=INVALID UNIT
```

Flush All Pages Command

The Flush All Pages command suspends accepting input data until all pages currently in the printer are printed. This gives the printer time to clear some memory.

$^{E}{}_{C}$ & r # F

\#   =0- Flush all complete pages
      1 - Flush all pages

Default = 0
Range = 0 - 1

A value of 0 indicates that only complete pages are to be processed. If a partial page exists, it is not processed. A value of 1 indicates that all page data including the partial page will be processed.

For example, if the printer contains two complete pages (page "A" and page "B") and one partial page (page "C"), and receives a Flush All Pages command (value field 0 - all complete pages), it ejects pages "A" and "B" and retains page "C." If the printer received the flush all pages (value field 1) it processes and ejects pages "A," "B," and "C."

The printer resumes receiving (processing) data when the last page is processed and ejected from the paper path.

Notes

Using the Flush All Pages command significantly reduces printing performance. If possible, applications should use the Free Space command to check for available memory without using the Flush All Pages command. If the memory status readback response indicates sufficient memory available to process a job, the Flush All Pages command does need not be used. If the memory response indicates insufficient memory available to run the job, then the application should use the Flush All Pages command to make memory available, and then check available memory a second time.

When possible, use the Flush All Pages command only at the beginning of a print job, prior to the receipt and processing of any data. This minimized, to minimize performance reduction.

Echo Command

The Echo command echoes its value field (in ASCII format) back to the host.

$^{E}{}_{C}$ * s # X

\#   =Echo value (ASCII)

Default **=** 0
Range **=** –32767 to 32767

If multiple users are requesting status, it can be difficult to distinguish one user's status response from another. The Echo command provides the means to label status responses. Since the user-selected value for the value field is returned, this command can be used as a user identification mark or "place holder." Sending the Echo command, with a specific user-selected value, at the beginning of a status request enables users to identify their status response data.

Notes

Status readback requests are processed in the order they are received.

Status readback responses are returned to the printer port from which they were received.

When selecting an echo value, it is important to select a number which is not likely to be used by another user, such as a random number.

This example illustrates why using the Echo command with a random number is important. Assume an application uses a fixed number each time the Echo command is sent. Further, assume the printer running the application was turned off after the application sent the Echo command followed by a Free Space command. The printer generates the status readback responses to the Echo command and Free Space commands. Since the host is not accepting data, the data will be in the printer waiting for the host to accept it. Now, assume the host computer is turned on and the application is again executed.

If the application sends the Echo command and Free Space command, the printer returns the response to the first Echo command and Free Space command, along with the response to the second request. The application will assume that the response received is the response to its last request. If the application uses random numbers in the Echo command, this type of mix-up will not occur.

Echo Response

The Echo command returns the following response:

```
PCL
ECHO ValueField
```

where “`ValueField`” is the Echo command value field value that was selected (within the range –32767 to 32767).

For example, if the Echo command, $^{E}{}_{C}$*s–999X was sent the status response would be:

```
PCL
ECHO –999
```

Status Readback Programming Hints

The following hints can assist in using the status readback feature.

- PCL status readback is useful during the development of applications. Status readback allows you to determine that fonts, macros, user-defined patterns, and symbol set resources you have downloaded were accepted by the printer.
- If the printer does not contain sufficient memory to accept a downloaded entity, the printer discards the data. Status readback can be used to determine if the printer accepted a downloaded entity.
- Status responses are directed to the printer's I/O port from which the request is received. If the status is not read and the printer switches to another I/O port, the status response is not directed to that port. The status response returns if the printer switches back to the original port (see note on following page).
- Your application should work correctly when an unexpected status response is received. For example, when requesting a PCL status, it is possible that PJL could return an unsolicited status response, if PJL is enabled.
    - All PCL status responses begin with the "PCL" header and end with the <FF> control code. When reading PCL status responses, your application should be able to read all the data between the "PCL" header and the <FF> control code. It should ignore any other status response syntax.
    - Lines within the PCL status response begin with a specific keyword (those described in this chapter) and end with the <CR> and <LF> control codes. Future printers may support new keywords in the PCL status response. Your application should be designed to ignore those lines which it does not understand.
- The first PCL status readback command an application should send is the PCL Echo command. A random number should be generated for the value field each time the command is sent. Your application should ignore all printer status readback data until the PCL Echo status readback response is received, echoing the number the application sent.
- To clear any possible unread status responses from previous applications, an application, upon starting up, may want to read any pending responses until they are cleared from the printer.